Cluster Computing manuscript No.
(will be inserted by the editor)

Early DGA-based botnet identification: pushing detection to the edges

Mattia Zago · Manuel Gil Pérez · Gregorio Martínez Pérez

Received: date / Accepted: date

Abstract With the first commercially available 5G infrastructures, worldwide's attention is shifting to the next generation of theorised technologies that might be finally deployable. In this context, the cybersecurity of edge equipment and end-devices must be a top priority as botnets see their spread remarkably increase. Most of them rely on algorithmically generated domain names (AGDs) to evade detection and remain shrouded from intrusion detection systems, via the so-called Domain Generation Algorithm (DGA). Despite the issue, by applying concepts such as distributed computing and federated learning, the cybersecurity community has prototyped and developed dynamic and scalable solutions that leverage the increased capabilities and connectivity of edge devices. This article proposes a lightweight and privacy-preserving framework that pushes the intelligence modules to the edges aiming to achieve early DGA-based botnet detection in mobile and edge-oriented scenarios. Experimental results prove the deployability of such architecture at all levels, including resource-constrained end-devices.

Keywords Domain Generation Algorithm (DGA) · Machine Learning · 5G · Cybersecurity · Edge Artificial Intelligence · Federated Learning

1 Introduction

As predicted [6], the past five years have seen the exponential growth of the research interest in 5G technology, and, nowadays, the first 5G commercial infrastructures are being deployed worldwide. Among other aspects and together with the newly improved service delivery requirements, the ultra-densification of connected devices forces scenarios in which fixed network architectures are not an option anymore [2]. Despite the numerous challenges that remain open [7], the research community has started to look beyond 5G [25].

A critical lesson that the community learned from the 5G research is that the service layer can be decoupled from the network architecture, resulting in frameworks that feature dynamic capabilities such as self-configuration, on-demand scalability, and self-protection. In such a scenario, Artificial Intelligence (AI) can be seen as a necessary construction block to sustain this required dynamicity. Indeed, the enabling technology that can offer the capabilities mentioned above consists of using the AI –Machine Learning (ML) and Deep Learning (DL)– to automate networks and services virtualisation, *e.g.*, AI for software-defined network (SDN) and network functions virtualization (NFV) self-optimisation. As a point of fact, several projects and vendors (5G America [1] and SELFNET [10] among others) proved that intelligent and scalable platforms based on ML-powered SDN/NFV could meet the exponentially increasing demands.

Not surprisingly, with this automation grade, the cybersecurity feature as a key principle across the heterogeneous solutions. Even though numerous frameworks have been developed to provide the required automation, security aspects are often overlooked [7]. To provide an example, proactive and reactive Intrusion Detection Systems (IDSs) appear to be limited to supervised analysis of network flows [7,15,24].To be precise, and to the best of our knowledge, there are a few high-grade network IDSs that include machine learning capabilities for other purposes than analysing network flows

Authors are with the Department of Information and Communications Engineering, University of Murcia, 30100 Murcia, Spain
E-mail: {mattia.zago,mgilperez,gregorio}@um.es

[7]. However, a clear trend is the widespread adoption of ML solutions at the farthest edges of the networks. Indeed, the concept of pushing computation towards the users is considered as one of the enabling technologies of the 21st century [12,26]. To be precise, both the concepts of AI-for-Edge and AI-on-Edge [12] have been widely explored before under different names, such as Multi-access Edge Computing (MEC), Mobile Cloud Computing (MCC), Transparent Computing (TC), Fog Computing, cloudlet [26]. All of them employed virtualisation techniques (*e.g.*, SDN/NFV) to decouple the services from the underlying hardware and features AI to optimise them automatically.

Notwithstanding the deployed analysis techniques, researchers agree on the urgency of tackling botnets and Advanced Persistent Threats (APTs), and specifically to identify their communication channels before the actual attacks can take place [21,37,39]. Within this context, the usage of Domain Generation Algorithms (DGAs) appeared as a clear trend due to the asymmetrical efforts required to sinkhole the generated domains [37,39]. The paper at hand will focus on ML applications for tackling these communication channels by providing the means to identify DGA-based botnets at scale.

The subject of detecting algorithmically generated domains (AGDs) is, in fact, offering a fertile research topic from multiple standpoints:

(i) firstly, the inherit randomness of the Fully Qualified Domain Names (FQDNs) makes static rule-based IDSs ineffective in favour of ML solutions;
(ii) secondly, the amount of FQDNs to be analysed compels researchers to develop scalable architectures;
(iii) thirdly, browsing history is to be considered a private subject, thus requiring privacy-aware solutions; and,
(iv) finally, it has been proved that APTs can stay dormant for prolonged periods before performing malicious actions, thus compelling preemptive detection capabilities.

In summary, the proposed approach leverages the well-known ideas that support the SDN/NFV paradigm to provide automated and scalable detection and reaction capabilities –also known as security-as-a-service (SECaaS) [40]. To be precise, the proposed framework demonstrates the capabilities of DGA-based botnet detection services deployed on the farthest edges of the network.

Hence, the fivefold contributions of the paper at hand can be outlined as follows.

- Firstly, this research will identify and discuss the key principles of Edge AI applied to ML-based network security;
- secondly, the literary works on DGA-based botnet detection are examined and mapped to the architectural designs that characterise Edge AI, eventually providing a comparison with pros and cons for each architectural design;
- thirdly, after a constructive discussion regarding the SECaaS deployment location, an experimental framework architecture is drafted;
- fourthly, a set of experiments will prove both the soundness of the Edge AI approach and the effectiveness of lightweight and explainable traditional ML algorithms in identifying DGA-based botnets; and,
- finally, the key ideas and principles identified throughout the research are blended in a lesson learned and future work discussion.

The rest of the paper at hand is structured as follows. Firstly, Section 2 will report the necessary background in terms of both Edge AI architectures and applications, with a specific focus on the difference between deployment locations; secondly, Section 3 will present the proposed prototypical framework for DGA-based botnet detection on edge. Then, Section 4 will gather and discuss some critical aspects with particular attention to future research objectives, while, finally, Section 5 will provide a conclusive summary.

2 Edge AI to look beyond 5G

To look beyond 5G technology is necessary to consider the enabling technologies that make the 5G ecosystem working [3]. Among them, the most important ones are undoubtedly the SDN, the NFV, the orchestration frameworks, and the containerisation theory [7]. Although Beyond 5G (B5G) heavily relies on wireless technology improvements [41], they are out of the scope for this research. Hence, instead of focusing on what makes B5G possible hardware-wise, this research looks at what architectural solutions and frameworks can be used.

Across the scale, the trend is clear: AI can and should automate and optimise most of these steps [12].

In this research, the main focus is on the early detection of DGA-based botnets in scenarios that features large volumes of data and a high degree of user mobility. As previously shown by [37], both classical ML (*i.e.*, those algorithms and models that do not employ neural networks) and DL solutions have been deployed to tackle this malware threat.

In the context of 5G and B5G networks, where potentially billions of devices are susceptible to malware infections, the ability to detect DGA-based botnets before they activate to attack other systems is critical. Edge AI offers a promising set of tools to study, design, and deploy scalable ad effective detection solutions.

Hence, using the DGA-based botnet detection as a use case, the remaining of this section is structured to introduce the necessary background on Edge AI (Section 2.1), the different architectural designs (Section 2.2), and their relations with the cybersecurity aspects (Section 2.3).

2.1 On the subject of Edge AI

As thoroughly surveyed by several authors [12,26,32,43], the Edge AI discipline encompasses all those techniques intended to move the ML (with great focus on DL ones) towards the end users instead of the cloud. In this research, we adhere to the definitions proposed by [43] that classifies the approaches to Edge AI as layers of a pyramid. Starting from the Cloud scenario, each subsequent level moves a part of the process towards the edges, having in the highest tier (Level 6) the whole process performed within the end devices. To be precise, with device it is identified any potential device that presents the properties of being of users' propriety and usage. Thus, in this category, falls both personal and company-issued devices such as laptops and smartphones, but also IoT devices such as routers, cameras, sensors.

As described in this section, Fig. 1 will describe and discuss the properties and differences of these six approaches (compared to the well-known cloud approach).

An entirely different subject is the application of AI to optimise and self-configure the network slices that provide the services, a topic defined as *Intelligence for Edge* or *AI for Edge* [12,32]. In the same surveys, the authors reported a collection of technologies designed to work at the edge of the networks; we suggest the readers refer to them, alongside with the numerous surveys in the area [11,18,19,23,26,33], for detailed information regarding the subject.

The SECaaS theory provides an example of this semantic difference. Generally speaking, a network IDS, such as the DGA-based botnet detection framework inhere presented, might be configured as a service using any virtualisation technology. Within the 5G ecosystem, this IDS would be designed as virtual network function (VNF) to be deployed as other services by the orchestrator. Similarly to 5G orchestration-level intelligence, AI for Edge [12] encompasses those ML applications that provide optimisation and learning capabilities to the management modules. On the contrary, AI on Edge covers the study of what kind of intelligence should be deployed in the IDS service, not how to deploy it; *e.g.*, how to separate malicious AGDs from legitimate FQDNs.

As for 5G orchestration, the location of the deployment of the service does influence the performances, especially in the context of the depicted use case of DGA-based botnet SECaaS. Thus, to highlight the differences between the approaches, in Fig. 1 are reported the prototypical architectures for each level in regards to the training (indicated with a red diamond marked with the TR acronym) and inference (indicated with a blue triangle with the INF acronym) phases. In this novel computation paradigm, low requirements tasks can be executed at the edges of the network, often directly on the end devices.

On the one hand, classical approaches rely upon the cloud datacenters for training the AI models [32], *i.e.*, the most resource-consuming part of the process. Whether it is possible to perform the training phase on the edges –and ultimately on the end-devices– heavily depends, among others, on:

(i) the use case, as not all scenarios have critical issues that can be mitigated by offloading the intelligence to the edges;
(ii) the latency and delay requirements, as, depending on the use case, there might be some constraints on the privacy requirements, *e.g.*, personalised user experience that strictly requires that no user data leave the device [16];
(iii) the isolation requirements as coexisting applications, cryptographical restrictions, and network slicing limitations may cause privacy and data issues;
(iv) the amount of data to be processed, *e.g.*, a face recognition service that needs to process hours upon hours of multiple feeds necessitate a non-trivial amount of bandwidths;
(v) the resource constraints of the end-devices, especially in the Internet of Things (IoT) ecosystem where the issues related to battery usage, broadband connection unreliability are more noticeable.

On the other hand, generally speaking, the data originate at the most remote edge of the network, *i.e.*, in the end-devices. As the data needs to be processed and inferred by the AI models, it is reasonable to aim at completing as much computation as possible directly on the devices (Levels 3 and 6 of Figure 1) or eventually offloaded to the edge (Levels 2, 4, and 5). Nonetheless, moving ML components to the device has some limitations, mainly regarding limited energy, computing capabilities, and storage [13].

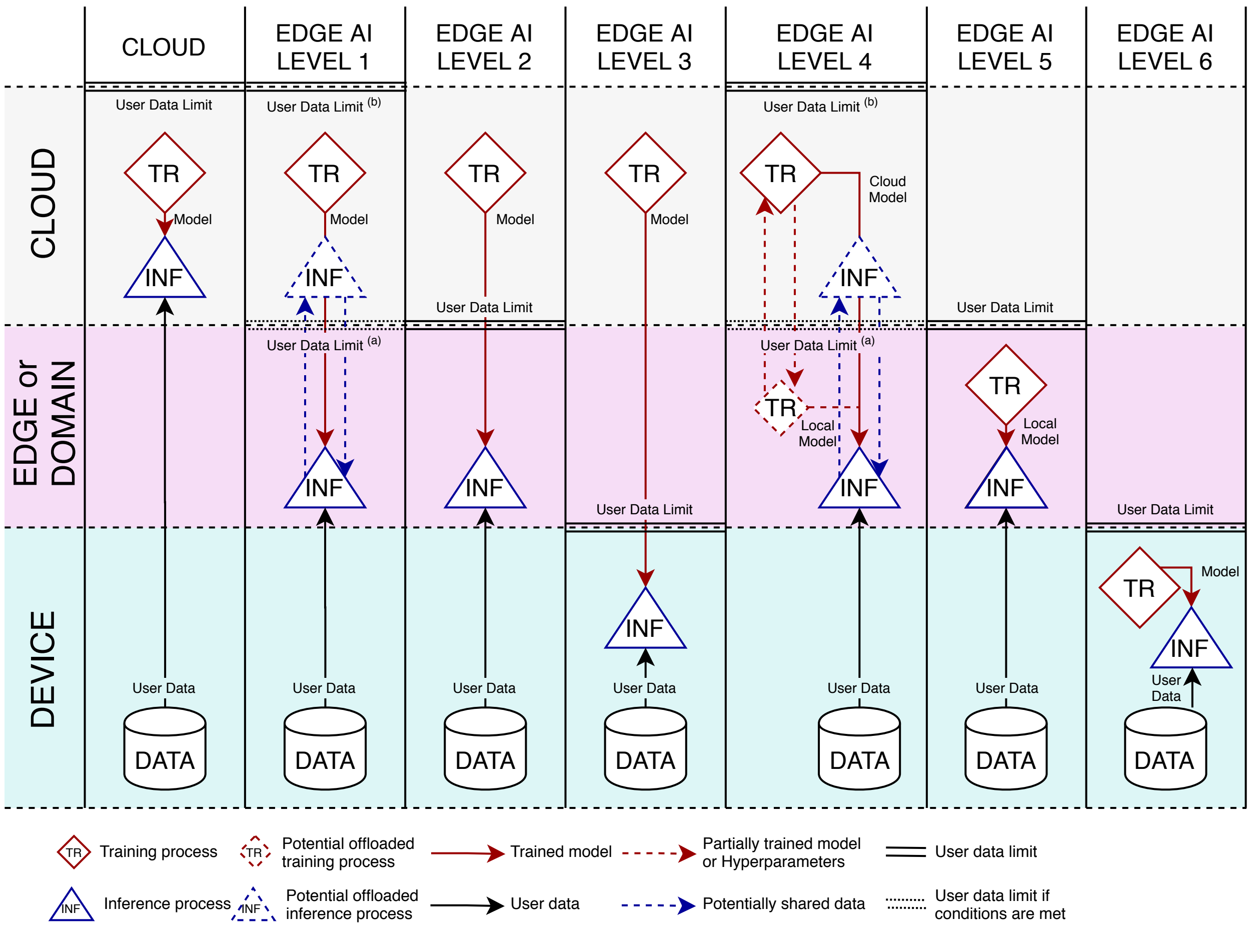


Fig. 1: Comparison overview for Edge AI protypical architectures.

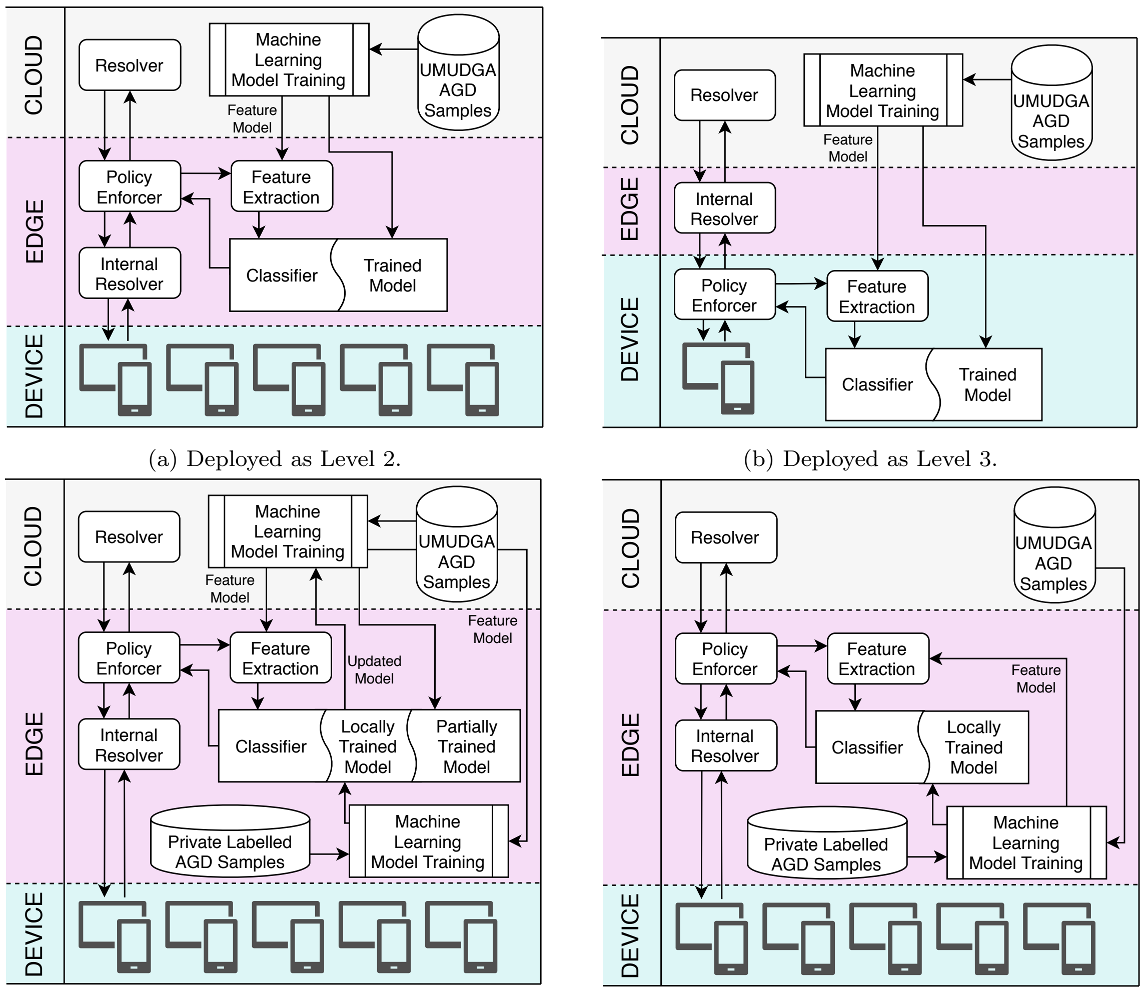
a/b User data limit when the inference process is carried out at the edge level (a) or at the cloud level (b).

Finally, and referring to Fig. 1, it is worth mentioning that only two levels, namely the third and the sixth, guarantee that the users' data never leaves the device. In Fig. 1, such limit is indicated with a double line, that can be either continuous or dotted. As will be described later on, in this last case, it represents the actual data limit in the case where no cloud-offloading is activated. Similarly, in the second and fifth levels, the inference process is performed on edge. Since these services are deployed closer to the users, there is a significant reduction in both latency and bandwidth; however, the security implications of leaving the data at the edge level are still unclear [18,22,23]. To be precise, if multiple edges merge their data to the cloud, privacy issues might arise, *i.e.*, the privacy concerns gathering around the extensive use of AI on end-devices are not related to the devices themselves, but with how and where the data are transmitted, processed and stored. Nonetheless, one of the main advantages of the centralised cloud relies on this uploaded and shared knowledge. In a collaborative environment such as has been studied before [27], IDSs can benefit from the intelligence gathered from multiple sources. Specifically, collaborative, cloud-based IDSs have been proved effective against zero day (0-day) malwares [8].

2.2 Architectural Differences

With regards to the different options presented in Fig. 1, this section will introduce a brief analysis of the most engaging aspects. To do so, we will introduce the Fig. 2 that presents a vertical analysis for Levels 2, 3, 4, and 5. To be precise, this section will unfold the options and capabilities of a DGA-based detection framework to be deployed as a collection of SECaaSs.

To start with the inference process location, Fig. 2a and Fig. 2b present two edges configurations. To begin with, Fig. 2a reports a prototypical architecture for a 5G-like network slice that interposes between the internal resolver and the remote ones. In this scenario, the



(a) Deployed as Level 2.

(b) Deployed as Level 3.

(c) Deployed as Level 4.

(d) Deployed as Level 5.

Fig. 2: Architecture for a DGA-based botnet detection framework deployed at different levels of Edge AI framework.

network slice is configured to apply a policy enforcer whose rules are defined by, for example, a ML classifier. Similarly, Fig. 2b reports a prototypical architecture for a on-device resolver that intercepts user's requests before sending them. In both configurations, the requested FQDN is extracted from the DNS query, processed by the feature extraction microservice, and classified using a pre-trained-cloud model. The difference between Fig. 2a and Fig. 2b dwells in the actual location of this process, *i.e.*, the edge and the end-device. Notice that both configurations require a trained model, which has been conveniently trained in the cloud. Mind also that there are no obligations to share the inferred data with the cloud services; thus, both the isolation and the privacy requirements can be achieved.

Similarly, Level 4 (Fig. 2c) and Level 5 (Fig. 2d) achieve the same isolation property by analysing the data directly on edge, without offloading it to the cloud. To better discuss it, it is necessary to refer back to Fig. 1, and specifically Levels 1 and 4. In both cases, the inference process takes place on edge; however, the architecture does leave open the possibility of offloading such tasks to the cloud services if needed. To put it in other terms, since the edges are generally resource-constrained, they might decide to offload part of the inference to the cloud services to optimise the workload. Offloading strategies include partial offloading, vertical collaboration, and horizontal collaboration, to cite a few (the readers might refer to [32] for a detailed survey on the subject). The main difference between the two approaches resides in the possibilities offered by transferring part of the training to the edges. That is to say, in Level 4, each edge can update the model and eventually share it with the cloud in a federated fashion. Even though the fourth level could be, in theory, designed to transfer the data to the cloud for load balancing purposes, in the scenario depicted in Fig. 2c, this is not the case. The classifier is designed to work only on edge; thus, the data limit can be considered on the edge boundaries, and not on the cloud ones.

So far, both Fig. 2a, Fig. 2b, and Fig. 2c have the training phase deployed as a cloud SECaaS. By contrast, in Fig. 2d the same process is configured to have the training phase on edge. To be precise, in Fig. 2d the in-edge training process uses a cloud-collected dataset that is augmented with private and locally-available data. It is worth mentioning that the training process does not require both local and cloud samples. However, shared knowledge [22, 27] is effective in tackling cybersecurity threats, specifically in the case of 0-days malwares. Although the scenario described in Fig. 2d offers a collaborative base for sharing knowledge, it does not include the actual share of the trained model (or its hyperparameters) as Fig. 2c (Level 4) does. In Fig. 2c, we assume that the edges are cooperating to produce an optimal model without sharing users' data, thus enforcing the isolation and privacy requirements. On the contrary, in Fig. 2d, the edges cannot share the user data to the cloud, hence forcing the central cloud to obtain its data somewhere else. In any case, a hybrid architecture is achievable, where multiple edges adopt a different model depending on local requirements: for example, in a context such the one offered by the DGA-based botnet detection, sharing users' browsing history might not be feasible without substantial anonymisation. Nonetheless, as proved in [37, 39], sharing a list of domain names (both legitimate FQDNs and AGDs) does not constitute a privacy issue.

2.2.1 Outlined differences, pros and cons

To summarise the different approaches, Table 1 reports their properties, advantages and disadvantages when the architectural models are applied to the DGA-based botnet detection.

For this comparison, it is worth mentioning that the difference between a 5G-style edge environment and a corporate domain one is of utmost importance, being it key to understanding the potential risks and benefits associated with cloud data [29]. In this scenario, several assumptions are, in fact, reasonable. Besides small and micro-enterprises, it is plausible to consider a private cloud environment, perhaps with a dedicated IT team or even a security operation center (SOC) in case of larger corporations. The resources available to these entities are not the same as those at the disposal of 5G-RAN nodes, albeit they share some advantages and disadvantages of Edge AI. Firstly, from the computation resources available, cloud training and testing feature virtually infinite resources; however, farthest edge pieces of equipment have limited means while end-devices have substantial constraints. Not surprisingly, latency heavily depends on the proximity to the users, the closest to them, the fastest the response. The concept of data source availability is deeply connected to the sharing framework between the participants, in this context, such as presented by [22], the cloud can either receive aggregated and anonymised data or a selection of parameters to update accordingly, ultimately providing a virtually unlimited overview of the network. The economic resources available at the highest levels also justifies the assumption of high-quality labelled data originating from the ones shared by the edges. The same happens within medium to large corporations, where a dedicated SOC might be providing curated data for the AI processes to work. For the same reason, data access can be limited to corporation assets, thus providing a similar isolation level as edge devices; as widely stated before in literature, once the data reaches the cloud, there is no guarantee besides what the service provider declared in the term of service (ToS).

In general, the strength of deploying a DGA-based botnet detection on edge relies on providing broad protection services (due to the collaborative efforts [27]) closer to the user, provided that the security triad (Confidentiality, Integrity, and Availability) of the data is guaranteed. However, as for every other collaborative framework, defensive mechanisms against data and model poisoning must be considered [34].

Finally, in terms of computational requirements for Edge AI, Khan *et al.* [19] reported a curated syllabus on expected technologies that are, in general, required for enabling Edge AI, while Coppolino *et al.* [11] specifically discussed hardware-based enhancement to make Edge AI feasible. Furthermore, Lin *et al.* [22] presented a comprehensive survey on actual implementations approaches, architectures and libraries to enable Edge AI in a decentralised and collaborative fashion.

2.2.2 Critical aspects of services location

To summarize, cloud services provide, by far, the most common scenario. In this context, a broadly employed–and thus lacking in innovation–approach is to propose frameworks that can provide SECaaS through self-deployable VNF [4, 40]. For these functionalities, most authors across the board agree on primary key performance indicators (KPIs) such as detection rates, yield capabilities, and guaranteed availability, indeed, the cloud scenario is characterised by scalable and high-reliable VNF that provide the required services to massive amounts of customers. Amid the criticalities, there are the privacy issues, well discussed in the past [20], that arise with sharing the data to third parties (first and foremost the

Metric		**DGA-based SECaaS deployment location**			
		Cloud	**Corporate Domain Private Cloud**	**Edge**	**End Device**
Computational resources availability		Unlimited	Depends on corporate resources	Limited	Constrained by power, battery and performance
Latency		Potentially high	Significantly high	Low	Zero
Data source availability	*Labelled*	Significantly high if in a collaborative framework	Depends on the size, scope, and availability of SOC.	None	None
	Not Labelled	Potentially unlimited if in a collaborative framework	Limited to corporate nodes	Limited to node	Limited to device
Data access and isolation		Third party accessible and variable with ToS	Corporate	Service provider	User
Benefits from shared intelligence	*Receiving data or models*	Multiple data sources, early identification of new threats	Broad Protection		
	Sharing data or models	Higher protection for the framework participants	Might get economical benefit from sharing data or models	Directly none. Indirectly, it contributes to the ecosystem	
Risks from shared intelligence	*Receiving data or models*	Data or model poisoning			
	Sharing data or models	Security CIA non-compliance, potentially higher risk of exposure due to higher data value			Security CIA non-compliance

Table 1: Qualitative comparison of different architectural approaches for a DGA-based detection framework compatible with Edge AI.

GDPR-related issues). Nevertheless, aggregating and collecting a massive amount of data can provide useful information to CERT and nation-wise monitoring actors without harming users' privacy.

With regards to DGA-botnet detection, the scenario itself does not offer any relevant challenge besides the ones already identified and explored in literature. Although supported by practically unlimited resources, SECaaS at cloud level suffers from scalability issues in regards to the number of connected devices, a critical aspect for B5G ecosystems.

Similarly to the cloud services, a few authors have designed SECaaS bearing in mind the potentialities of edge computing, for example, by pushing the VNFs to the remotest areas of the networks. Regarding the KPIs, and besides those already defined for the cloud services, authors have collected and defined several metrics based on the hardware requirements needed for running the VNF. Although some authors make the distinction between the different categories of KPI indicators (*e.g.*, quality of service (QoS) and quality of experience (QoE) among others), such a discussion falls outside of the scope of this research. Indeed, this research field hankers for a precise and formal definition of metrics and indicators to enable a quantitative comparison between reproducible frameworks. To cite an example of such indicators, there is the capacity of utilisation, *i.e.*, the percentage of classification capacity used over a predefined time unit that enables optimised load balancing for network intrusion detection [15].

To further discuss the applicability of a DGA-based detection module at the edge level, it is necessary to separate the concept of edge and corporate domain.

Edge computing – On the one hand, the edge-related discussion should pivot on the automated, human-free, capabilities, highlighting themes such as the limited computational resources, the extremely low-latency requirements and the native isolation from the cloud services.

Corporate domain – On the other hand, corporate domains can be partially overlapped with the concept of the private cloud, thus centring the discussion on themes such as the protection and isolation of the sensible data rather than fully automated detection capabilities. The private cloud environment provides mitigation against the already mentioned privacy issues of the cloud providers, without tackling the benefits of the broad protection that a shared knowledge base can provide [9].

Provided that the corporate domain can guarantee the security CIA compliance, and as previously reported in Table 1, the scenario can present several benefits to the SECaaS, specifically in terms of the amount of data and overall network visibility. In comparison, a fully automated on-edge configuration would permit much faster responses due to the reduced latency and high self-capabilities. However, a dedicated SOC, if available, can be handy in detecting anomalies and new APTs, thus providing constant updates and improvements to the detection models.

Lastly, only a few authors have proposed detection solutions that work entirely on the end-devices [14,23]. In this constrained scenario, hardware related KPIs becomes critical, *e.g.*, execution time and resource consumption. Bear in mind that the end-device category does not only include smartphones and personal devices, but it also encompasses a broad category of IoT devices such as home routers and Industrial Internet of Thingss (IIoTs) sensors. Indeed, Edge AI solutions proliferate in industrial scenarios due to the low resources, high availability constraints [35].

2.3 Edge AI and cybersecurity

There are a few notable solutions worth mentioning concerning the cybersecurity side of the applications compatible with the Edge AI paradigm.

Besides the inherited threat model from the cloud, edge computing introduces new security risks due to its traits [23]. Ideally, a fully privacy-preserving approach would only manage strictly necessary data without transmitting them to any external processing centre (in Fig. 1, only levels three and six achieves this status). We invite the readers to refer to [12,22,32] for technical analysis of up-to-date libraries to deploy frameworks in such fashion.

On the attack side of the cybersecurity, Isakov *et al.* reported an in-depth analysis of the current state of the art regarding the threats [18]. In their survey, they firstly introduced a taxonomy on vulnerabilities and criticalities of neural networks that can be exploited by criminals to gain an advantage on the cyber defences; also Liu *et al.* [23] reported a collection of security threats that target data explicitly.

On the defensive side, several aspects need to be taken into account. First and foremost, most (if not all) data have protection and isolation requirements; consequently, the security challenges associated with those subjects are of the highest importance and priority. In such a sense, several surveys have collected, analysed, and compared specific mitigation techniques to the vulnerabilities that have been gradually uncovered. Among them, Liu *et al.* [23] analysed the phases of collection, processing, and storage of the data management on the edges, with particular attention to the open challenges and future research directions.

For another aspect of the defensive side, classical ML powered detection frameworks are evolving toward decentralised, edge-oriented solutions. Two scenarios prevail among others when looking at the research interest; on the one hand, the industrial environment offers requirements for high-availability, low-resources and low-latency services. The derived research challenges cause the research community to thrive [35,42]. On the other hand, the challenges and requirements offered by the 5G self-protection scenarios mainly lead the community to design IDSs as full-cloud services [24], having just some of them offloading part of the inference process to the edges [15,17]. In that sense, [14] stands out by presenting a working solution for a Level 6 (fully on-device) architecture for IoT devices. Notably, mixed solutions like the ones proposed by the federated learning paradigm [22,33], *i.e.*, those solutions where the training is at least partially performed on edge are hyped and are notably worth further researches.

Within network detection, to the best of our knowledge, this is the first attempt to explore DGA detection on the edges. Notably, several frameworks have been proposed just in the last year [5,28,31,36], however, besides apparent reproducibility issues [39], every work allegedly resolves the challenges related to DGA-based botnet detection. Nevertheless, there is a clear trend in terms of the chosen technology, *i.e.*, Long Short-Term Memory network (LSTM)-based and, in general, DL-powered framework [28,31,36].

In addition to generic network-based malware detection, a particular focus should be dedicated to the APT threat [21]. These advanced malwares are often recompiled for the specific target and present multiple obfuscated variants that do not match classical signatures. However, the extensive usage of machine learning in log and data mining has been proved useful to detect malware infection symptoms, especially new ones (0-day). In the context of this manuscript, the same ideas are applied to the network communication phase [39] to identify a botnet communication as early as its first connection to the Command & Control (C&C). In the context of Edge AI and, in general B5G, it appears that this threat explodes with the number of newly connected and poorly protected devices [35].

Last but not least, the distributed nature of these architectures requires to discuss the underling security primitives that guarantee the confidentiality and integrity of the shared data [29]. However, subjects like trust and reputation management alongside with access control, authentication and encryption will not be discussed in this research article as they are out of scope and they have already been widely covered in literature [29].

3 Experimental DGA-based botnet detection on Edge

Most of the Edge AI reported in the previous sections offer little-to-no architectural challenges besides the one offered by implementing the actual algorithms on the resource-constrained edge devices. Despite that, the Federated Learning approach offers an intermediate solution that connects the best cloud performances with the isolation and scalability offered by the edge characteristics. The remaining of this section is structured as follows: Section 3.1 depicts the framework's architecture, while Section 3.2 presents a brief remark in terms of minimum computational resources and modules locations.

3.1 Proposed architecture

The framework described in the paper at hand is designed to be compatible with the Edge AI Level 4 in a Federated Learning fashion. Fig. 3 presents its architecture, including both the cloud level services and two different edge-level environments leveraging the idea of a collaborative framework The edges, represented as isolated domains, receive a shared and pre-trained detection model that can be augmented with locally available data. However, albeit it is possible to improve the model in such a way using the same data structure (*e.g.*, feature set and format, among others) it is also true the opposite. Locally poisoned data could skew the model in an adversarial fashion [34,30]. Similarly, each domain can willingly provide data or partial models to be included in the shared cloud next training iteration. Bear in mind that, as will be demonstrated by the experimental results, the whole edge plan might be shifted to an end-device, provided that the libraries for executing the code are available. To the best of our knowledge, this is the first attempt to establish a DGA-based botnet detection framework that can be deployed this far in the edges.

At a high level, Fig. 3 presents three areas, namely the Cloud level (on the top half, with a white background) and two edge levels (on the bottom half, with yellow and blue backgrounds). Edges are also identified as "domains" due to the extendibility of the proposed framework to an enterprise scenario. In such a case, the company infrastructure might be composed of several edges that rely on a single shared training subcomponent.

One of the crucial advantages of having enforced the separation of the detection modules relies on the data and model isolation [22]. The cloud components can be seen as multiple SECaaS provided by a cybersecurity vendor, while each edge/domain represents a federated subscriber. In Fig. 3, "Domain A" does not share model nor data to the cloud, simulating a restricted environment where the collected user data cannot be pushed to the cloud. No information regarding eventual APTs and 0-days are thus available for the federation. On the contrary, "Domain B" does share the pieces of evidence and samples of collected malwares, enabling the cloud model to be updated with the new information. Instead of data, the hyperparameters of the model could be shared to provide an extra isolation level [22].

Among the various levels identified in Fig. 1 and discussed in Section 2, we considered Level 4 as the one that provides both the flexibility of deployment required by automated 5G scenarios and the compatibility with the isolation requirements characteristics of corporate environments. To be precise, as will be discussed in Section 3.2, the whole process could be executed directly on the end-devices (thus achieving the Level 6 compatibility); however, we deemed more interesting to evaluate the potential flexibility offered by the federated learning environment, *i.e.*, a collaborative network of virtualised and lightweight SECaaS to provide DGA-based botnet detection at scale.

In this configuration, the two domains represent the duality of the Level 4 architecture design presented in Fig. 1. The user data limit is, indeed, different. In fact, for domain "Domain A", that does not share user data with the central cloud, the limit is enforced at the boundaries of the edge. Independently from the location of the classification process, "Domain B" does share data with the cloud provider, thus moving the limit mentioned above to the boundaries of the cloud services. To summarise, the isolation properties heavily depend on the actual deployment configuration and application scenarios; policies and restrictions might apply to different use cases, as well as risks and benefits.

In terms of the chosen technology, besides several DL algorithms that can be used for DGA-based botnet detection [28,31,36,37], we will adhere to the explainable AI philosophy in the current manuscript. Indeed, although DL solutions are trending, this research will empirically demonstrate that classical and lightweight models can achieve excellent results. Albeit Federated Learning focuses on DL, the same principles can also apply to some classical models, first and foremost the tree-based ones. As a consequence, all the experiments will be carried out with decision-tree based solutions like the Random Forest algorithm or modern approaches such as the XGBoost or LightGBM.

The remainder of this section has been divided into three parts to ease the framework exploration process; firstly, the data are described and presented in Section

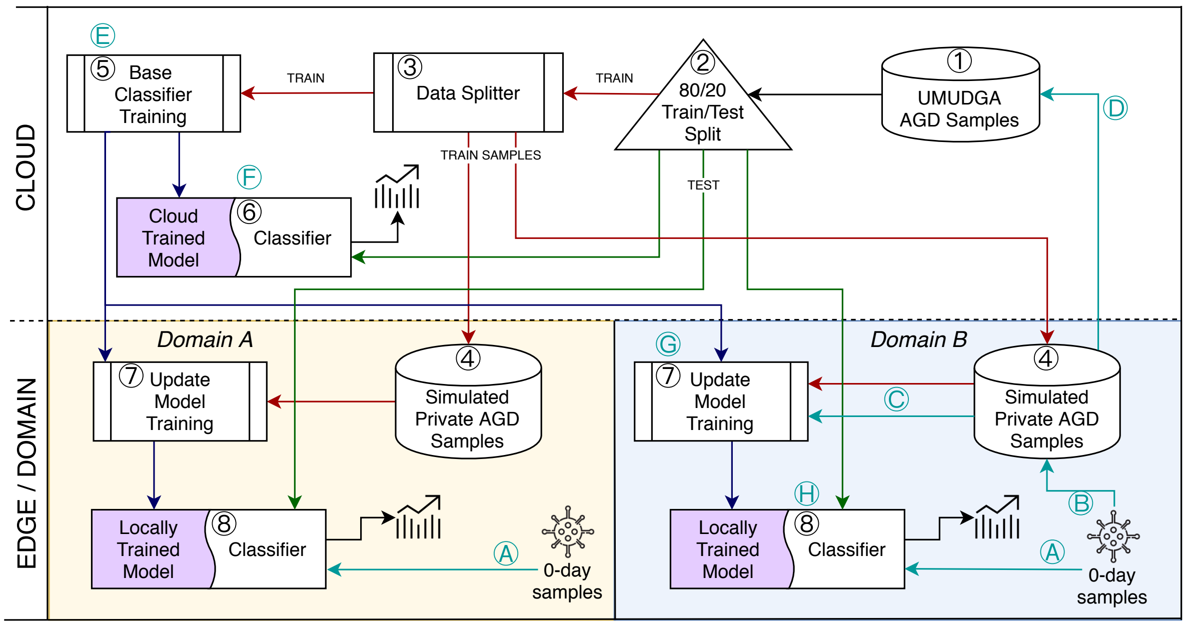


Fig. 3: Architecture and data flow for the proposed L4-compatible experiment.

3.1.1; then, the primary framework's loop (*i.e.*, steps from ① to ⑧) is analysed in Section 3.1.2; finally, an example of the feedback loop (*i.e.*, steps from Ⓐ to Ⓗ) is provided in Section 3.1.3 in the advent of a new 0-day infection.

3.1.1 Preprocessing and feature analysis

The proposed experiment uses the data collected by the authors [39] and publicly released in [38]. In the dataset, 50 malware variants have been collected and described, providing both the raw lists of AGDs and a preview of the extracted features. Nevertheless, not all reported features provide enough information to be used for detection purposes [37,39]. Hence, a feature selection process has been carried out to limit the overheat provided by the curse of dimensionality, also given the context of low-resource or resource-constrained devices provided by the Edge AI scenario [37]. Furthermore, some malware classes have been grouped due to their indistinguishability [39] with a combination of clustering techniques and careful human revision. Of the 50 malware families identified, 21 clusters have been identified. Fig. 4 reports the classes with the associated clusters. The clustered data have been resampled (200,000 FQDNs for the training set, and 9628 FQDNs for the testing set, both stratified).

Features are ranked with a recursive feature elimination process, eliminating one feature per iteration. Experimental results suggest that the top 10 features are representative enough to achieve good classification results. Table 2 reports these classification performances for the three classifiers picked for the analysis. By combining the information reported in Table 2 with the one reported in Table 3 it is possible to notice that the average resource consumption halves by accepting a 2% loss in F1 score. The proposed trade-off enables scalable and dynamic reconfiguration of the detection model, but simply switching to the most reactive and less computationally expensive model depending on the traffic volumes. To be precise, Table 3 reports the results in terms of resource consumption's for both the full feature set and the top 10 feature sets.

Features	**Algorithm**	**F1 Micro**	**F1 Macro**
Full	Random Forest	0.9523	0.9514
Top 10	Random Forest	0.9399	0.9386
Full	LightGBM	0.9612	0.9604
Top 10	LightGBM	0.9473	0.9459
Full	XGBoost	0.9024	0.8988
Top 10	XGBoost	0.8794	0.8749

Table 2: Model performances comparison for feature selection.

3.1.2 Data flow and experimental results

For the experiments, the framework samples 10,000 domain names for each provided class obtained from [38], and the framework is built upon the Random Forest classifier with the warm start option enabled to allow the in-edge upgrade of the model.

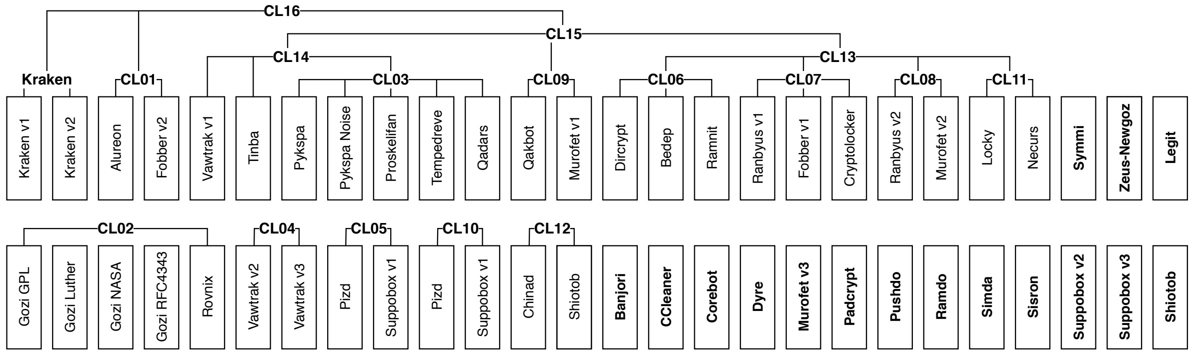


Fig. 4: Partial Hierarchical Cluster of UMUDGA [38] classes, all the non-connected boxes are considered standalone classes.

Feat.	Algorithm	Type	Total Time	Instance Time	Memory Peak	Memory Increment
Full	RF	Train	75.243	0.00038	1157.18	395.82
Full	RF	Test	0.630	0.00007	1066.05	12.23
Top 10	RF	Train	31.555	0.00016	1115.76	363.69
Top 10	RF	Test	0.651	0.00007	1124.62	12.27
Full	LightGBM	Train	305.845	0.00153	1377.81	641.18
Full	LightGBM	Test	2.436	0.00025	757.02	0.34
Top 10	LightGBM	Train	45.275	0.00023	768.75	9.84
Top 10	LightGBM	Test	1.627	0.00017	769.01	0.26
Full	XGBoost	Train	1669.997	0.00835	1543.91	595.18
Full	XGBoost	Test	1.529	0.00016	1345.97	0.31
Top 10	XGBoost	Train	261.434	0.00131	757.41	2.48
Top 10	XGBoost	Test	1.411	0.00015	757.49	0.08

Table 3: Model resources consumption comparison for feature selection (duration are in seconds, memory usage is in MiB).

The data flows from a central shared data source, indicated in Fig. 3 with the number ① to the edge classifiers indicated with the number ⑧. The elements marked with circled letters are instead analysed later in Section 3.1.3.

A standard 80/20 separation is adopted for splitting the data in step ②. On the one hand, the resulting testing set (20%) will be used at cloud and edge levels to ensure comparability results. On the other hand, the training data (80%) is shuffled and separated into three different, possibly stratified sets with configurable proportions in ③. The subcomponent separates the data to simulate data that are available only at the edges level ④.

The cloud training module ⑤ will use the first data set to train the base model ⑥ to be shared with the edges. The first experimental results are gathered at this phase by evaluating this classifier model against the testing set and reported in Table 4.

The base model is then shared with all edges belonging to the federation and augmented with locally collected data ⑦; these data are simulated by sharing one training set with the edge in step ③. The updated model is evaluated in ⑧ against the same testing data used in the previous phase; the results are also available in Table 4. From the table perspective, it is possible to notice that the results provided by updating the model ⑦ and validating it against the testing set ⑧ do not necessarily lead to an improvement in the model. In other words, retraining the model with additional data might not result in an increased detection ratio [30]. Nevertheless, this local-update functionality is of undoubted interest, and future researches might consider it when exploring security architectures.

3.1.3 Simulating a Zero-Day APT

One of the critical aspects of having ML-powered detection modules is the capability of identifying suspicious behaviours never seen before. This experiment aims to simulate the framework data flow in case of a new malware (which could be easily interpreted as an early stage APT infection).

While referring to Fig. 3), this experiment flow is identified by circled letters (*i.e.*, from Ⓐ to Ⓗ) instead

Phase	Trees	Classes	Samples	0-Day	F1 Micro	F1 Macro	Confidence
⑤ Cloud Training	50	21	127,761	✗	0.929	0.928	0.957
⑥ Cloud Testing	50	21	39,926	✗	0.931	0.930	0.910
⑦ Edge Retrain	50 +10	21	159,702	✗	0.931	0.930	0.935
⑧ Edge Test	50 +10	21	39,926	✗	0.931	0.929	0.906
Ⓐ Edge Test Zero Day	50 +10	21	8,000	✓	0.776	0.855	0.812
Ⓔ Cloud Retrain with Zero Day	50	22	128,761	✓	0.930	0.931	0.957
Ⓕ Cloud Testing	50 +10	22	47,926	✓	0.943	0.933	0.925
Ⓖ Edge Retrain	50 +10	22	187,725	✓	0.929	0.929	0.934
Ⓗ Edge Testing	50 +10	22	47,927	✓	0.943	0.937	0.923

Table 4: Random Forest performances for the simulated experiment, before and after injecting the zero day.

Role	Component	Time (s)									Med.	STD
Server	⑤ Fitting	65.05	60.55	79.05	64.70	76.26	64.04	64.03	63.81	63.56	64.04	6.33
	⑤ 10 CV	236.30	235.92	274.03	13.20	14.35	13.44	15.02	13.85	14.49	14.49	117.86
	⑥ Validation	0.31	0.31	0.35	0.31	0.35	0.32	0.70	0.33	0.33	0.33	0.13
Client w/ retrain	Load Model	0.10	0.10	0.15	0.09	0.14	0.12	0.12	0.15	0.10	0.12	0.02
	⑦ Add. Fitting	2.73	2.78	3.01	2.80	3.07	2.91	2.94	2.90	2.93	2.91	0.11
	⑧ Validation	0.37	0.36	0.41	0.45	0.41	0.37	0.46	0.48	0.47	0.41	0.05
Client w/o retrain	Load Model	0.08	0.11	0.11	0.12	0.13	0.13	0.14	0.13	0.10	0.12	0.02
	⑧ Validation	0.32	0.31	0.35	0.34	0.37	0.34	0.34	0.31	0.32	0.34	0.02
	Processors	**1**	**1**	**1**	**2**	**2**	**2**	**4**	**4**	**4**		
	Memory	**1GB**	**2GB**	**4GB**	**1GB**	**2GB**	**4GB**	**1GB**	**2GB**	**4GB**		

(a) Time requirements

Role	Component	Memory Usage (MB)									Med.	STD
Server	⑤ Fitting	251	251	251	266	266	267	294	294	299	266	20
	⑤ 10 CV	677	677	677	806	827	825	764	917	970	806	106
	⑥ Validation	427	426	426	540	561	558	470	624	671	540	91
Client w/ retrain	Load Model	410	410	410	409	410	410	409	410	410	410	0
	⑦ Add. Fitting	410	410	410	427	424	427	435	432	434	427	11
	⑧ Validation	410	410	410	432	430	431	442	437	442	431	14
Client w/o retrain	Load Model	410	410	410	410	409	410	410	410	410	410	0
	⑧ Validation	410	410	410	410	410	411	417	417	418	410	4
	Processors	**1**	**1**	**1**	**2**	**2**	**2**	**4**	**4**	**4**		
	Memory	**1GB**	**2GB**	**4GB**	**1GB**	**2GB**	**4GB**	**1GB**	**2GB**	**4GB**		

(b) Memory requirements

Table 5: Performances of the machine learning processes depending on the resources dedicated to the virtualised environment

of numbers. In step Ⓐ, a few AGD samples are injected in the analysed set of data to simulate a previously unseen malware establishing a communication channel with the C&C.

Two scenarios are available at this point, *i.e.*, depending on the Edge concept's interpretation, some capabilities might or might not be enabled. On the one hand, in "Domain A", (that simulates a classic, fully automated, 5G-RAN environment) the new threat will remain unnoticed. On the other hand, in "Domain B" (that simulates a corporate domain with active monitoring tools), the SOC eventually notices the confidence decrease (see Table 4, row Ⓐ Edge Test Zero Day) and investigate the matter. In this last scenario, some samples will be eventually collected Ⓑ and used to update the local model Ⓒ. Possibly, and in a federated learning fashion, these data could be shared and aggregated at the cloud level Ⓓ, to be studied (Ⓔ and Ⓕ), and eventually distributed to some edges Ⓖ as the base model. Similarly to the forward-loop (steps ①-⑧), the results are reported in Table 4.

3.2 Resources requirements

A computation analysis has been carried out to evaluate the various components capabilities and requirements. The results, presented in Table 5, encompass both the time (Table 5a) and the memory (Table 5b) requirements of the subcomponents. In the table, the training process indicated in Fig. 3 as ⑤ has been divided into the initial generic fitting and the optional cross-validation process. Similarly, the clients have been divided accordingly to the option availability of a retraining component (indicated in Fig. 3 as ⑦).

To begin with the experiment configuration, a unique VNF has been configured to execute each submodule independently. On the subject of resources allocated, several typical profiles have been taken into account, ranging from 1 dedicated core and 1 GB SDRAM (typical configuration of a Raspberry PI IoT device) to 4 dedicated cores and 4 GB SDRAM (typical configuration of a low-end personal laptop). With the specific configuration of the chosen ML model, lowering the minimum memory resources is not possible. Nevertheless, new specialised IoT-oriented ML libraries might provide comparable detection performances while operating under lower resources requirements. Both the results reported in Table 5 and previously in Table 3 demonstrate that even without optimised mobile-specific ML libraries it is possible to achieve excellent and explainable detection results in a resource-constrained environment.

4 Lessons learned and future works

Edge AI represents an innovative solution to several high-ends cybersecurity issues. While local models excel at keeping the information isolated from others, they also require a non-negligible amount of shared data to be able to target threats effectively.

In other words, while predictive models such as [16] works well by only using the individual user data, other detection models–such as the one proposed in this research–are not suitable to work individually. To be precise, detection models benefit from the shared knowledge base gathered either at cloud level or collaboratively in an edge-federated fashion; local models are great to learn the particular context in which they are deployed, but they miss the broad vision that only the cloud can provide. Indeed, some scenarios might require extensive computations for preprocessing before the actual inference process kicks-in; for example, in the context of IIoT, most of the sensors deployed do not have the minimum resources (including battery capacity) to perform any intensive computation [35]. For the DGA-based botnet detection, tackling the malwares at the DNS level enables to deploy lightweight platform-independent probes capable of privacy-aware real-time inspections at scale. Future researches should include the tradeoff between performances, latency, and privacy aspects in the architectural design.

Concerning the data protection subject, future researches might focus on ways to aggregate and anonymise data (*e.g.*, homomorphic and searchable encryption), knowledge transfer learning, gossip training, as well as explainable ML (and DL) models which hyperparameters could be shared within the collaborative network. The parameter sharing approach could potentially remove a substantial amount of computation and still provide a powerful detection suite. In this sense, future works might include detectors capable of preemptively block connections that are suspected of belonging to botnet networks by merely analysing the DNS queries.

On the subject of traditional ML vs DL solutions, we deem that a remark is needed: DL is not required in every domain. In the depicted scenario, the DGA data does not feature complex non-linear relationships. As established before [37,39] and remarked in Section 3, a small number of features will suffice to train classical ML algorithms achieving good results. As empirically demonstrated in the section, a fully on-device DGA-based botnet detection is possible with traditional ML algorithms, given that enough samples for each class are provided. Moreover, most approaches focus on supervised learning as it provides relatively straightforward and verifiable pipelines; despite their clear benefit, supervised solutions require large datasets and careful data supervision. Unsupervised approaches have also been explored, although not in-depth [37], and future researches might focus on analysing semi-supervised hybrid solutions to take advantage of the massive amounts of unlabelled data.

For another key lesson, attacks at collaborative models are not something new [30], as a such, DGAs are going to evolve to mimick legitimate FQDNs and potentially to tackle the detection algorithms directly in an adversarial fashion. On-edge solutions, and precisely in a federated learning ecosystem, have been proved susceptible to adversarial attacks [22]. In such a sense, future works might discuss whether edge computing is necessary, desired, or even feasible.

Finally, as previously indicated by the authors [37, 39], the field in-here studied features a general lack of reproducibility:

– First and foremost, the data used to power the frameworks are rarely shared; therefore, future researches should make use of already shared and well-known datasets, or provide their own with the appropriate comparisons to the state-of-art.
– Secondly, the deployed models are described, but not released (often neglecting to comment on the hyperparameters configuration); as such, future researches should focus on producing reproducible results that can be tested and validated by the community.
– Thirdly, although quantitative comparison frameworks for ML (and DL) algorithms do exist, their application is often limited to aggregated indexes

that might deceive the results, *e.g.*, the Area Under the Curve (AUC); hence, future researchers should report all the outcomes, especially the negative ones.
– Lastly, the actual implementations are often described and tested outside a proper validation framework; in such a sense, future researches might focus on developing a suite of tools and indicators to enable formal comparisons.

5 Conclusions

Albeit the disruptive innovation led by the 5G enabling technologies, researchers did not stop in exploring more advanced solutions. Among others, Edge AI is believed to be the next enabling technology for what is coming beyond the 5G. It is in this direction that collaborative concepts, such as federated learning, empowers the raw potential of 5G bandwidth and number of devices, combining it with lightweight but highly effective machine learning models. On-device AI might, in some cases, represent the perfect solution for data-sensible scenarios; however, Edge AI aspires to serve as an intermediate compromise between cloud practically unlimited resources and privacy constraints. With this scenario in mind, this research empirically proves that a DGA-based botnet detection module is not only feasible to be deployed in the resources-constrained environment, but it also would benefit from shared intelligence in a federated fashion. A natural evolution of this research would be to extend the detection models also to tackle frequent domain name attacks such as the typosquatting (*i.e.*, URL hijacking) or to identify less-known techniques such as the DNS-based data exfiltration.

Acknowledgements This work has been supported by a predoctoral grant from the Spanish National Cybersecurity Institute (INCIBE) within the program "Ayudas para la Excelencia de los Equipos de Investigación Avanzada en Ciberseguridad" ("Grants for the Excellence of Advanced Cybersecurity Research Teams") with code `INCIBEI-2015-27353`, the European Commission Horizon 2020 Programme under grant agreement number `H2020-SU-DS-2019/883335` - PALANTIR (Practical Autonomous Cyberhealth for resilient SMEs & Microenterprises), and the European Commission (FEDER/ERDF).

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. 5G Americas: 5G at the edge. Tech. rep., 5G Americas (2019)
2. 5G PPP Architecture Working Group: View on 5G architecture. Tech. rep., 5G Infrastructure Public Private Partnership (2020). DOI 10.5281/zenodo.3265031
3. Afolabi, I., Taleb, T., Samdanis, K., Ksentini, A., Flinck, H.: Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. IEEE Communications Surveys and Tutorials **20**(3), 2429–2453 (2018). DOI 10.1109/COMST.2018.2815638
4. Ahmad, I., Shahabuddin, S., Kumar, T., Okwuibe, J., Gurtov, A., Ylianttila, M.: Security for 5G and beyond. IEEE Communications Surveys and Tutorials **21**(4), 3682–3722 (2019). DOI 10.1109/COMST.2019.2916180
5. Almashhadani, A.O., Kaiiali, M., Carlin, D., Sezer, S.: MaldomDetector: A system for detecting algorithmically generated domain names with machine learning. Computers and Security **93**, 101787 (2020). DOI 10.1016/j.cose.2020.101787
6. Andrews, J.G., Buzzi, S., Choi, W., Hanly, S.V., Lozano, A., Soong, A.C.K., Zhang, J.C.: What will 5G be? IEEE Journal on Selected Areas in Communications **32**(6), 1065–1082 (2014). DOI 10.1109/JSAC.2014.2328098
7. Barakabitze, A.A., Ahmad, A., Mijumbi, R., Hines, A.: 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges. Computer Networks **167**, 106984 (2020). DOI 10.1016/j.comnet.2019.106984
8. Blaise, A., Bouet, M., Conan, V., Secci, S.: Detection of zero-day attacks: An unsupervised port-based approach. Computer Networks **180**(January), 107391 (2020). DOI 10.1016/j.comnet.2020.107391
9. Chadwick, D.W., Fan, W., Costantino, G., de Lemos, R., Di Cerbo, F., Herwono, I., Manea, M., Mori, P., Sajjad, A., Wang, X.S.: A cloud-edge based data security architecture for sharing and analysing cyber threat information. Future Generation Computer Systems **102**, 710–722 (2020). DOI 10.1016/j.future.2019.06.026
10. Chirivella-Perez, E., Marco-Alaez, R., Hita, A., Serrano, A., Alcaraz Calero, J.M., Wang, Q., Neves, P.M., Bernini, G., Koutsopoulos, K., Gil Pérez, M., Martínez Pérez, G., Barros, M.J., Gavras, A.: SELFNET 5G mobile edge computing infrastructure: Design and prototyping. Software: Practice and Experience **50**(5), 741–756 (2020). DOI 10.1002/spe.2681
11. Coppolino, L., D'Antonio, S., Mazzeo, G., Romano, L.: A comprehensive survey of hardware-assisted security: From the edge to the cloud. Internet of Things **6**, 100055 (2019). DOI 10.1016/j.iot.2019.100055
12. Deng, S., Zhao, H., Fang, W., Yin, J., Dustdar, S., Zomaya, A.Y.: Edge intelligence: The confluence of edge computing and artificial intelligence. IEEE Internet of Things Journal pp. 1–13 (2020). DOI 10.1109/jiot.2020.2984887
13. Dhar, S., Guo, J., Liu, J., Tripathi, S., Kurup, U., Shah, M.: On-Device Machine Learning: An Algorithms and Learning Theory Perspective (2019). URL `http://arxiv.org/abs/1911.00623`
14. Eskandari, M., Janjua, Z.H., Vecchio, M., Antonelli, F.: Passban IDS: An intelligent anomaly based intrusion detection system for IoT edge devices. IEEE Internet of Things Journal pp. 1–16 (2020). DOI 10.1109/JIOT.2020.2970501
15. Fernández Maimó, L., Perales Gómez, Á.L., García Clemente, F.J., Gil Pérez, M., Martínez Pérez, G.: A self-adaptive deep learning-based system for anomaly detection in 5G networks. IEEE Access **6**, 7700–7712 (2018). DOI 10.1109/ACCESS.2018.2803446

16. Hard, A., Kiddon, C.M., Ramage, D., Beaufays, F., Eichner, H., Rao, K., Mathews, R., Augenstein, S.: Federated learning for mobile keyboard prediction (2018). URL `https://arxiv.org/abs/1811.03604`
17. Huertas Celdrán, A., Gil Pérez, M., García Clemente, F.J., Martínez Pérez, G.: Towards the autonomous provision of self-protection capabilities in 5G networks. Journal of Ambient Intelligence and Humanized Computing **10**(12), 4707–4720 (2019). DOI 10.1007/s12652-018-0848-6
18. Isakov, M., Gadepally, V., Gettings, K.M., Kinsy, M.A.: Survey of attacks and defenses on edge-deployed neural networks. In: 2019 IEEE High Performance Extreme Computing Conference, pp. 1–8 (2019). DOI 10.1109/HPEC.2019.8916519
19. Khan, W.Z., Ahmed, E., Hakak, S., Yaqoob, I., Ahmed, A.: Edge computing: A survey. Future Generation Computer Systems **97**, 219–235 (2019). DOI 10.1016/j.future.2019.02.050
20. Kumar, R., Goyal, R.: On cloud security requirements, threats, vulnerabilities and countermeasures: A survey. Computer Science Review **33**, 1–48 (2019). DOI 10.1016/j.cosrev.2019.05.002
21. Laurenza, G., Lazzeretti, R., Mazzotti, L.: Malware Triage for Early Identification of Advanced Persistent Threat Activities. Digital Threats: Research and Practice **1**(3), 1–17 (2020). DOI 10.1145/3386581
22. Lim, W.Y.B., Luong, N.C., Hoang, D.T., Jiao, Y., Liang, Y., Yang, Q., Niyato, D., Miao, C.: Federated learning in mobile edge networks: A comprehensive survey. IEEE Communications Surveys Tutorials pp. 1–33 (2020). DOI 10.1109/COMST.2020.2986024
23. Liu, D., Yan, Z., Ding, W., Atiquzzaman, M.: A survey on secure data analytics in edge computing. IEEE Internet of Things Journal **6**(3), 4946–4967 (2019). DOI 10.1109/JIOT.2019.2897619
24. Papamartzivanos, D., Gomez Marmol, F., Kambourakis, G.: Introducing deep learning self-adaptive misuse network intrusion detection systems. IEEE Access **7**, 13546–13560 (2019). DOI 10.1109/ACCESS.2019.2893871
25. Pham, Q.V., Fang, F., Ha, V.N., Piran, M.J., Le, M., Le, L.B., Hwang, W.J., Ding, Z.: A survey of multi-access edge computing in 5G and beyond: Fundamentals, technology integration, and state-of-the-art. IEEE Access **8**, 116974–117017 (2020). DOI 10.1109/ACCESS.2020.3001277
26. Ren, J., Zhang, D., He, S., Zhang, Y., Li, T.: A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet. ACM Computing Surveys **52**(6), 125 (2019). DOI 10.1145/3362031
27. Sharma, R., Chan, C.A., Leckie, C.: Evaluation of Centralised vs Distributed Collaborative Intrusion Detection Systems in Multi-Access Edge Computing. In: 2020 IFIP Networking Conference (Networking), pp. 343–351 (2020)
28. Shi, W.C., Sun, H.M.: DeepBot: A time-based botnet detection with deep learning. Soft Computing **61** (2020). DOI 10.1007/s00500-020-04963-z
29. Sun, P.J.: Privacy protection and data security in cloud computing: A survey, challenges, and solutions. IEEE Access **7**, 147420–147452 (2019). DOI 10.1109/ACCESS.2019.2946185
30. Tabassi, E., Burns, K.J., Hadjimichael, M., Molina-Markham, A.D., Sexton, J.T.: A taxonomy and terminology of adversarial machine learning - DRAFT. Tech. rep., NIST (2019). DOI 10.6028/NIST.IR.8269-draft
31. Tian, J., Gou, G., Liu, C., Chen, Y., Xiong, G., Li, Z.: DLchain: A covert channel over blockchain based on dynamic labels. In: Information and Communications Security, pp. 814–830 (2020). DOI 10.1007/978-3-030-41579-2_47
32. Wang, X., Han, Y., Leung, V.C., Niyato, D., Yan, X., Chen, X.: Convergence of edge computing and deep learning: A comprehensive survey. IEEE Communications Surveys & Tutorials pp. 869–904 (2020). DOI 10.1109/comst.2020.2970550
33. Wang, X., Han, Y., Wang, C., Zhao, Q., Chen, X., Chen, M.: In-Edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning. IEEE Network **33**(5), 156–165 (2019). DOI 10.1109/MNET.2019.1800286
34. Xu, X., Liu, X., Yin, X., Wang, S., Qi, Q., Qi, L.: Privacy-aware offloading for training tasks of generative adversarial network in edge computing. Information Sciences **532**, 1–15 (2020). DOI 10.1016/j.ins.2020.04.026
35. Yao, H., Gao, P., Zhang, P., Wang, J., Jiang, C., Lu, L.: Hybrid intrusion detection system for edge-based IIoT relying on machine-learning-aided detection. IEEE Network **33**(5), 75–81 (2019). DOI 10.1109/MNET.001.1800479
36. Yun, X., Huang, J., Wang, Y., Zang, T., Zhou, Y., Zhang, Y.: Khaos: An adversarial neural network DGA with high anti-detection ability. IEEE Transactions on Information Forensics and Security **15**, 2225–2240 (2020). DOI 10.1109/TIFS.2019.2960647
37. Zago, M., Gil Pérez, M., Martínez Pérez, G.: Scalable detection of botnets based on DGA: Efficient feature discovery process in machine learning techniques. Soft Computing **24**, 5517–5537 (2020). DOI 10.1007/s00500-018-03703-8
38. Zago, M., Gil Pérez, M., Martínez Pérez, G.: UMUDGA: A dataset for profiling algorithmically generated domain names in botnet detection. Data in Brief **30**, 105400 (2020). DOI 10.1016/j.dib.2020.105400
39. Zago, M., Gil Pérez, M., Martínez Pérez, G.: UMUDGA: A dataset for profiling DGA-based botnet. Computers & Security **92**, 101719 (2020). DOI 10.1016/j.cose.2020.101719
40. Zargar, S.T., Takabi, H., Iyer, J.: Security-as-a-Service (SECaaS) in the cloud. In: Security, Privacy, and Digital Forensics in the Cloud, chap. 9, pp. 189–200. John Wiley & Sons, Ltd (2019). DOI 10.1002/9781119053385.ch9
41. Zhang, C., Ueng, Y., Studer, C., Burg, A.: Artificial Intelligence for 5G and Beyond 5G: Implementations, Algorithms, and Optimizations. IEEE Journal on Emerging and Selected Topics in Circuits and Systems **10**(2), 149–163 (2020). DOI 10.1109/JETCAS.2020.3000103
42. Zhang, Y., Huang, H., Yang, L.X., Xiang, Y., Li, M.: Serious challenges and potential solutions for the industrial internet of things with edge intelligence. IEEE Network **33**(5), 41–45 (2019). DOI 10.1109/MNET.001.1800478
43. Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., Zhang, J.: Edge intelligence: Paving the last mile of artificial intelligence with edge computing. Proceedings of the IEEE **107**(8), 1738–1762 (2019). DOI 10.1109/JPROC.2019.2918951