The Evolution of Federated Learning-based Intrusion Detection and Mitigation: a Survey

Léo Lavaur, Marc-Oliver Pahl, Yann Busnel, Fabien Autrel

The Evolution of Federated Learning-based Intrusion Detection and Mitigation: a Survey

Léo Lavaur
IMT Atlantique, IRISA, Cyber CNI
leo.lavaur@imt-atlantique.fr

Marc-Oliver Pahl
IMT Atlantique, IRISA, Cyber CNI
marc-oliver.pahl@imt-atlantique.fr

Yann Busnel
IMT Atlantique, IRISA
yann.busnel@imt-atlantique.fr

Fabien Autrel
IMT Atlantique, IRISA
fabien.autrel@imt-atlantique.fr

***Abstract*—In 2016, Google introduced the concept of Federated Learning (FL), enabling collaborative Machine Learning (ML). FL does not share local data but ML models, offering applications in diverse domains. This paper focuses on the application of FL to Intrusion Detection Systems (IDSs). There, common criteria to compare existing solutions are missing. In particular, this survey shows: (i) how FL-based IDSs are used in different domains; (ii) what differences exist between architectures; (iii) the state of the art of FL-based IDS.**

With a structured literature survey, this work identifies the relevant state of the art in FL–based intrusion detection from its creation in 2016 until 2021. It provides a reference architecture and a taxonomy to serve as guidelines to compare and design FL-based IDSs. Both are validated with the existing works. Finally, it identifies research directions for the application of FL to intrusion detection systems.

***Index Terms*—federated learning, machine learning, intrusion detection systems, collaborative sharing, network security management, attack mitigation**

I. Introduction

Modern information security has become complex. It faces the interconnection of heterogeneous networks, device types, protocols, and objectives [1]. This complexity threatens both Information Technology (IT) and Operational Technology (OT) infrastructures. Collaboration can help to cope with cyberattacks. For instance, technical and contextual Threat Intelligence (TI) sharing enables raising cyber-awareness [2]. Moreover, regulations evolve to promote and encourage coordination and collaboration in security [3]–[5]. To feed alerting and sharing mechanisms, Intrusion Detection Systems (IDSs) monitor networks and systems to detect attacks. These processes are slow and complex, while actionable intelligence requires timeliness [6]. Furthermore, information-sharing can lead to privacy and confidentiality issues [2].

The introduction of Federated Learning (FL) in 2016 by Google [7] enabled its application in multiple domains, such as intrusion detection. Since then, the popularity of the IDS use case has increased in the literature. Using FL enables local detection and mitigation with low latency, while collaboratively learning from others [8], [9] while preserving privacy [10], [11]. FL also promises to solve other drawbacks of state-of-the-art Machine Learning (ML)–based IDS, *e.g.* local bias due to a lack of heterogeneity in the training dataset [12]. Compared with Collaborative IDS (CIDS) approaches, FL reduces latency and bandwidth issues [13]. But federating local models also introduces new challenges, such as reputation and trust, computing resources availability, or data distribution.

Several existing works propose FL-based intrusion detection approaches [9], [12]–[32]. Intrusion detection is a critical aspect of modern security with specific constraints. For instance, detection systems need to provide alerts fast enough to be able to react [25]. They also must deal with heterogeneous network settings and devices [13], and cope with unknown attacks [21]. Therefore, it is necessary to identify the influence that have different federation settings on detection performance. Recent surveys propose to approach FL and intrusion detection as a joint topic [33], [34], highlighting the relevance of FL-based IDSs for the community. The term Federated Intrusion Detection System (FIDS) will be used throughout the document for FL-based IDS, with or without the mitigation aspect.

Related surveys [33]–[35] give an overview of FIDSs and existing approaches. However, the topic of FIDS still lacks structure and completeness. Consequently, this work sheds light on the topic of FIDS by performing a structured literature review. This includes qualitative and quantitative analyses, and establishes a taxonomy and a reference architecture for Federated Intrusion Detection Systems. First, this work extracts relevant classes from existing taxonomies on related topics: ML–based anomaly detection [36], CIDS [37] and FL [38]. Second, the comparison of the selected works enables the definition of comparison criteria that are added to the taxonomy. The selection of the reviewed works is validated by multiple steps: intuitive search, structured search, snowballing, venues and groups analysis.

The survey answers the following questions:

- How are FIDSs used in different domains?
- What are the differences between FIDS architectures?
- What is the state of the art of FIDSs?

The contributions of this paper are fourfold. (1) It reviews

This research is part of the chair CyberCNI.fr with support of the FEDER development fund of the Brittany region.

the application of FL to attack detection and mitigation. To do so, it reviews literature in the associated subtopics, using quantitative and qualitative approaches. (2) It proposes a reference architecture that generalizes selected works, and can serve as a starting point for designing future FIDS. (3) It establishes a taxonomy of federated detection and mitigation systems, which provides a framework for the comparison of the selected works in this survey. (4) It highlights open issues concerning FIDS, and identifies relevant research directions.

The paper is structured as follows. Section II presents the related works. Section III details the methodology, and states the research questions (RQs). Section IV introduces the domain and details the challenges FIDSs help to cope with. In Section V provides an analysis of the selected works quantitatively and qualitatively. Finally, Section VI discusses the outcomes and limitations of existing works, before proposing relevant research directions.

II. Related Work

Numerous surveys exist in literature on the topics of collaboration and intrusion detection [2], [33]–[35], [37]–[50]. However, only three of them address both, the collaborative aspects of FL and its application to intrusion detection and mitigation. The subject of FIDSs still lacks structure and completeness. Apart from FIDSs, reviewed related works can be categorized in three subtopics (see Section IV): (a) information sharing, (b) intrusion detection, (c) collaborative ML.

Table I summaries the selected papers, sorted according to their focus. The comparison criteria have been chosen to differentiate this work with related surveys. Criteria include focus and objectives of related works, sharing characteristics, and contributions. The topic coverage of a paper in the table is defined as follows:

- a topic is considered *covered* (●) when several references around the topic are cited and their outcomes are discussed;
- a topic is considered *partly covered* (◐) if at least one reference is cited, and its outcomes are explained;
- a topic is considered *not covered* (○) if the topic is either only referred to or not mentioned at all.

In the contributions, *qualitative* literature review refers to studying the content of selected works, possibly with a structured comparison, while a *quantitative* review extract objective numeric information such as years, or number of papers per domain, and draws conclusions. *Performance evaluation* relates to reimplementing all or part of selected works to compare their performances. Some surveys have been chosen for their number of citations on Google Scholar, the others because they represent the only related survey in their domain. Finally, in some domains such as Cyber Threat Intelligence (CTI) or Artificial Intelligence (AI) security, multiple surveys have been selected for the sake of completeness.

Common issues of collaborative sharing systems, such as trust and reputation, also apply to FL-based collaboration systems. Therefore, four surveys [2], [39]–[41] are included in this work to provide perspectives on the collaborative aspects of FL. The authors discuss the advantages and limitations of information-sharing (a), especially CTI. They highlight a need for standardization, automation, and incentives, in order to achieve efficient collaboration. The present survey differs by focusing on the technical aspects of automated collaboration.

As the topic of intrusion detection (b) is a critical part of FIDSs, this work reviews three related surveys on intrusion detection [43]–[45], especially relying on ML algorithms. They also discuss the usage of distributed ledgers, blockchain-based in particular, to support collaborative IDS. The blockchain is also one of the considered solutions to enable decentralized FIDSs, as shown in Section V-C6.

FL is the second critical aspect of FIDSs. Consequently, related works include surveys on the collaborative aspects of ML (c) and FL [47], [50]. They discuss FL approaches to work with distributed architectures. The security of FL is also heavily reviewed by Mothukuri *et al.* [38], Lyu *et al.* [48], and Shen *et al.* [49]. They identify security threats like communication bottleneck, poisoning, and Distributed Denial of Service (DDoS) attacks, that could endanger FL-based systems. While the IDS use case can be seen as an application of FL, we show it raises specific concerns in terms of privacy, latency, and adaptability.

Vasilomanolakis *et al.* [37] and Zhou *et al.* [46] survey the evolution of CIDS—at the merge of intrusion detection (b) and collaborative ML (c). Their works are however older and thus, cannot offer a comprehensive view of CIDS, as FL–based approaches did not exist at the time of their writing. Hence, the authors focus on collaboration in the sense of *detection+correlation*, whereas the following analysis (Section V-C) surveys the use of FL in IDSs.

Finally, recent work have reviewed the use of FL for intrusion detection [33]–[35]. Alazab *et al.* [35] address the wider topic of FL for cybersecurity, which only includes intrusion detection as an application. Their paper is explanatory and provides an overview of FL applications in information security. Like this work, Agrawal *et al.* [33] focus on FIDSs, but have different methodology. The authors list existing FIDSs and detail their approaches, and identify open issues. On the other hand, Campos *et al.* [34] review a subset of FIDSs by focusing on Internet of Things (IoT) use case, and the impact of non-IID (Independent and Identically Distributed) data on performance. While all identify challenges and research directions, this work also performs quantitative (Section V-B) and qualitative (Section V-C) analyses of existing FIDSs, and extracts reference architecture and taxonomy. The existence of these papers emphasizes the importance and relevance of FIDSs for the research community.

III. Methodology

This section details the methodology deployed to review the state of the art of FIDSs. This article follows the Systematic Literature Review (SLR) methodology [51]. SLR uses analytical methods to answer research questions about the literature on a specific topic. Existing SLR–based articles help to structure and to format this work, like [52] or [50].

TABLE I: Related works, their topics, contributions, and number of citations according to Google Scholar – Oct. 2021

Domain	Year	Reference	Security	Privacy	Trust	Defense/Mitigation	Detection	Cyber-awareness	Data	Models	Events	Indicators	Defensive measures	Qualitative literature review	Quantitative literature review	Taxonomy	Reference architecture	Performance evaluation	Research directions	Cited
Sharing—(a)	2016	Skopik *et al.* [39]	○	○	○	○	○	●	○	○	●	●	○	●	○	○	○	○	●	170
	2018	Tounsi *et al.* [40]	○	○	◐	○	○	●	○	○	○	●	○	●	●	○	○	○	●	181
	2019	Wagner *et al.* [2]	●	●	●	○	○	●	○	○	○	●	○	●	●	○	○	○	●	45
	2019	Pala *et al.* [41]	●	○	●	○	○	●	○	○	○	●	●	●	●	○	●	○	●	13
Detection—(b)	2016	Buczak *et al.* [42]	○	○	○	●	○	○	○	○	○	○	○	●	○	○	○	○	◐	1749
	2018	Meng *et al.* [43]	○	○	○	●	○	○	●	○	●	○	○	●	○	○	○	○	●	338
	2019	Chaabouni *et al.* [44]	○	○	○	●	●	○	○	○	○	○	○	●	○	◐	○	○	●	246
	2019	da Costa *et al.* [45]	○	○	○	●	○	○	○	○	○	○	○	●	○	○	○	○	●	152
Collaborative detection—(b) and (c)	2010	Zhou *et al.* [46]	◐	●	◐	●	○	○	●	○	●	○	○	●	○	○	○	○	●	474
	2015	Vasilomanolakis *et al.* [37]	◐	●	◐	●	○	○	●	○	●	○	○	●	○	●	○	○	●	270
FL—(c)	2020	Aledhari *et al.* [47]	●	●	◐	○	○	○	○	●	○	○	○	●	○	○	○	○	○	83
	2020	Lyu *et al.* [48]	●	●	○	○	○	○	○	●	○	○	○	●	○	○	○	○	●	101
	2020	Shen *et al.* [49]	●	●	◐	○	○	○	○	●	○	○	○	●	○	○	○	○	●	4
	2021	Mothukuri *et al.* [38]	●	●	○	○	○	○	○	●	○	○	○	●	○	●	○	○	●	81
	2021	Lo *et al.* [50]	●	●	○	○	○	○	○	●	○	○	○	●	●	○	○	○	●	18
FIDS	2021	Agrawal *et al.* [33]	●	●	◐	●	○	○	○	●	○	○	○	●	○	○	○	○	●	1
	2021	Alazab *et al.* [35]	●	●	●	◐	◐	◐	○	●	○	○	○	●	○	○	○	○	●	0
	2021	Campos *et al.* [34]	●	●	◐	●	○	○	○	●	○	○	○	●	○	○	○	●	●	0
	2022	**Lavaur *et al.***	●	●	◐	●	◐	○	○	●	○	○	○	●	○	○	●	●	○	●	-
			Focus			Objective			Sharing							Contributions					

● covers topic; ◐ partly addresses topic; ○ does not cover topic;

A. Research questions (RQs)

The SLR methodology recommends defining explicit research questions to structure the review and the selection of papers. This survey aims at evaluating FIDS and their maturity, as well as their core components, and relevant variations. Therefore, using related and selected works, we identify the following RQs that cover the topic of FIDSs:

(1) What are FIDS?

RQ 1.1. What challenges do FIDS help to cope with?

RQ 1.2. Which techniques exist to federate ML–based detection and mitigation mechanisms?

(2) What are the differences between of FIDS?

RQ 2.1. What are the key components of FIDS? How do they influence the system's performance?

RQ 2.2. Which metrics are used to measure and compare FIDS?

(3) What is the state of the art of FIDS?

RQ 3.1. What are the topics covered by the academic literature since 2016?

RQ 3.2. Where was the literatures published? Which research groups and communities are active in this area?

RQ 3.3. What are open questions according to existing works?

The RQs are organized in three categories. First comes the nature of FIDSs, and the existing approaches in Sections IV and V-C. Then, the survey analyzes what differentiates one system from another, and how the differences can be measured, in Section V-C. Finally, this work reviews the status of the literature on FIDS in Section V-B and Section VI.

B. Search and selection processes

Figure 1 presents the methodology and its search, selection, and synthesis processes. In yellow are represented the sources of papers, in green the final selection, and in gray the processing steps of the methodology. The tools used in the *Structured search* are represented with search engines in purple, and online databases in blue.

The searching of relevant literature involves four sources: recommendations, intuitive search, structured search, and snowballing.

(1) *Recommendations* were given by supervisors and coworkers throughout the realization of this work. This initial set of relevant papers is also used as a source of snowballing for further searching. Moreover, we included references from an aborted survey on *Collaborative security approaches*, which already yielded a substantial amount of literature by using the same methods.

(2) *Intuitive search* has been performed at the beginning of the survey to get a first grasp on the topic, and to learn about the functioning of FIDSs. At first, mostly Google Scholar has been used.

(3) *Structured search* has been adopted afterward, following the principles of SLR [51]. Different search engines and online databases are used for the sake of completeness, as illustrated in Figure 1. Databases can provide different results depending on their ownership. Search engine results differ according to the way requests are parsed, and the papers they have indexed. Thus, multiple sources provide more exhaustive results. The results can be reproduced by using the two *search strings* that were used in the

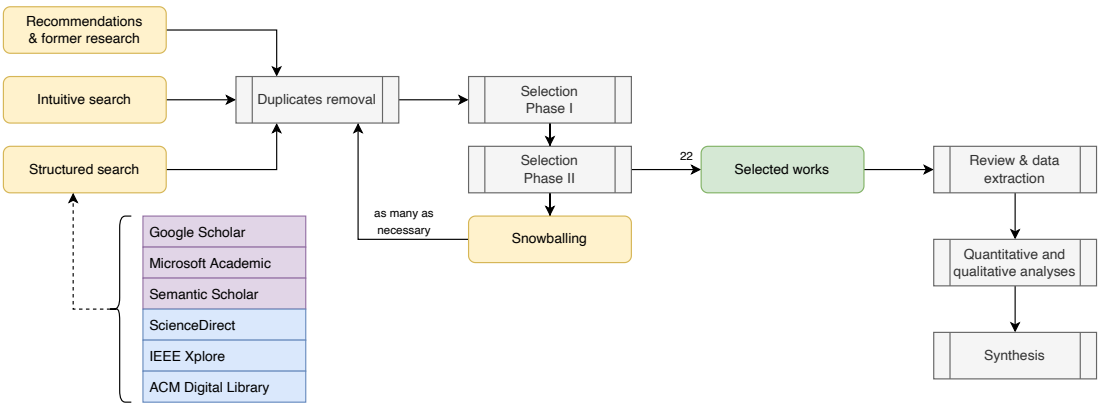


Fig. 1: Search and selection processes

structured search process: (a) application of FL to IDSs, and (b) literature addressing the topic of FIDS with unusual keywords.

(a) `("federated learning" OR "fl" OR "federated") AND ("intrusion detection systems" OR "ids")`

(b) `("federated" OR "collaborative") AND ("detection" OR "defense" OR "mitigation")`

(4) *Snowballing* identifies relevant works that would have been missed otherwise, such as publications cited by articles of our selected corpus, or papers that refer to them. The related surveys identified in this work (Section II) contain a lot of references to technical articles, making them relevant for snowballing. Furthermore, as this survey proceeds with quantitative analysis of the venues and groups (Section V-B), it provides extended snowballing opportunities by looking at other publications in the most represented venues or research groups in the selected corpus.

Approximately two hundred papers have been identified. Duplicate removal is performed with Zotero which allows identifying and merging redundant items. The selection then happens in two phases. Firstly, the title and abstract are used to discriminate *out-of-scope* papers in Phase I, along with their number of citations given the search engines, and age. However, a paper with few citations, but interesting abstract, probably only lacks visibility. Thus, it is moved to Phase II, which consists of a more thorough analysis of the selected works, using the *three-pass* approach defined by Keshav [53].

After the two selection phases, 22 papers were selected, excluding the 18 initial surveys seen in Section II. All present technical solution for FIDS. The challenges identified in Section IV were also used to either search or select papers, mostly through the *intuitive search* part.

C. Taxonomy and reference architecture

The qualitative analysis performed in this work section (V-C) provides a comparison of the selected works. We propose a taxonomy to identify the relevant criteria to differentiate the solutions. The synthesis is structured around the twelve classes of the taxonomy.

The taxonomy is built upon different existing taxonomies regarding CIDS [37], [46], ML–based intrusion detection [45], and FL [38], [47], [48]. First, we extract classes relevant to the domain of FIDS. In order to filter out irrelevant classes, the taxonomy is validated against the reference architecture (Figure 3). The proposed architecture displays both the operation and the design of the system. By confronting the taxonomy and the architecture, we ensure that each item of the taxonomy is related to a component of the architecture, and *vice versa*.

Then, the commonalities between the selected works that are not already represented in the previous taxonomies are added. This identifies new criteria on which to compare the selected works. Validation of both propositions is provided by the literature review, if each selected work can be studied through taxonomy items.

D. Metric selection

To review performance, we select common metrics for ML-based detection systems according to related surveys [36], [42], [44] (Section II), as well as other works on the topic [54], [55]. While these metrics are good indicators of the effectiveness of the selected systems, they cannot be used to compare them. As pointed out in Section V-C, differences in terms of datasets and architectures make comparison irrelevant.

In the selection, some works also consider other kinds of metrics, like CPU usage or network latency. However, they cannot be used to compare works either as they heavily rely on hardware choices. Actual comparison of selected works would require a complete reimplementation on common hardware and software stacks (Section VI-C).

Finally, selected metrics are sometimes used to compare FIDS with an *ideal* scenario, where models are trained on all data at once [9], [22]. Comparison with CIDS approach allows focusing on the federation and aggregation aspects of FIDSs (Section IV-C). The selected metrics are presented and explained in Section IV-D.

IV. Background

This section defines the concept of collaborative security and overviews its limitations. It first details the subtopics for intrusion detection (Section IV-A) and collaborative ML (Section IV-B). Then, it identifies the corresponding challenges that FIDSs cope with, answering the corresponding RQ (1.1). Section IV-C introduces FL, and motivates its use for intrusion detection. Finally, selected performance metrics are presented (Section IV-D).

A. Machine Learning for Intrusion Detection

To protect organizations, security systems often rely on signature-based IDSs to detect known attacks [44]. This approach is however inefficient against novel or zero-day attacks and Advanced Persistent Threats (APTs), like Stuxnet [44]. Furthermore, the heterogeneity and sporadic traffic of the IoT make IDSs less efficient or inadequate [12], [44]. Hence, the research community started to explore anomaly detection as a solution to improve detection systems [44].

Anomaly-based detection systems compare a normal profile, trained upon nominal traffic, with observed events to determine if they are malicious [56]. To that end, researchers started to study ML to identify the abnormal behavior using unsupervised learning. The other main approach is supervised classification. But supervised ML algorithms need labeled data to train upon. Table II sums up the most frequent datasets used to train ML-based IDSs.

For real-world use, algorithms must be trained on relevant data to perform well. This is a problem in IT, but especially in OT. IoT devices generate less traffic, producing more homogeneous training data in smaller quantities [12]. This issue is aggravated in siloed configurations, *i.e.* in which models are executed locally. Local architectures induce two obstacles. First, fewer data to train the models means that the collected dataset is less exhaustive, missing unseen samples [11]. This may cause models to yield more false positives or negatives.

Challenge 1. ML models trained on local data are less well-equipped against unknown behaviors.

Second, local data in real-world are likely to be collected on devices with little variance, *e.g.* same brand, same protocols, or use cases. Thus, there is a risk of generating a biased model, which would misclassify data, and eventually raise too many alerts [55].

Challenge 2. ML models trained on local data increase the risk of introducing bias.

Consequently, the siloed architecture of detection systems is an obstacle to their effectiveness [47] and IDSs can benefit from data-sharing through federation.

B. Collaborative ML

ML in the context of cybersecurity spans over multiple approaches and domains, from pattern recognition [64] to anomaly detection [12].

Prior to the advent of FL, most collaborative ML solutions had a centralized analysis, which induces its own set of issues. Centralized systems represent a single point-of-failure (SPoF) in the architecture [9]. This also applies to centralized iterations of CIDS before the introduction of FL. With FIDS however, analysis is done locally. Thus, even a failure on a federation server would not impede local detection.

Challenge 3. Typical CIDSs are centralized, and therefore represent a SPoF.

Moreover, centralized analysis means that all the data collected by the probes (Figure 3) must be assembled in a central location, and thus transferred. Data transfer implies increased latency and bandwidth. However, TI and alerts must also be actionable to be of use in an operational context. The ENISA, the European Union Agency for Cybersecurity, defines in [6] the actionability of TI as the fulfillment of five criteria: relevance, timeliness, accuracy, completeness, and ingestibility. Relevancy depends on the context of the recipient. Accuracy and completeness depend on the emitter, which is assumed to be exemplary in this context. Timeliness and ingestibility however are mostly provided by the supporting architecture. Hence, an analysis closer to the capture location would participate in decreasing latency and bandwidth consumption [11].

Challenge 4. Centralized CIDSs increase latency and bandwidth when compared to local detection.

Furthermore, data transfer can represent a privacy risk for a company, as the data relevant for intrusion detection is likely to contain sensitive information [46]. Exposed information might reveal relevant insights for an attacker.

Challenge 5. Centralized CIDSs can expose sensitive information.

Collaboration might imply unverified participants—*i.e.* whom trustworthiness cannot be asserted. Stakeholders might then be reluctant to involve their organization in sharing processes [2]. While FIDSs do not overcome this issue by design, research is required in this direction, as noted in Section VI-C4.

C. Federated Learning

Konečný, McMahan, Yu, *et al.* [65] introduced FL in 2016, originally to reduce the communication overhead induced by data sharing; the technology has since then been studied intensely (Section V-B1). FL works by aggregating the models generated by on-device training to benefit from the experience of other participants without sharing local data. Aggregation can be performed by a server [12], [18], [66], [67] which can induce trust and privacy issues. Recent research tends toward the use of trusted distributed ledgers such as blockchain to improve availability, traceability, and integrity [20], [68].

FL can also vary depending on the objectives on the federation. Cross-device Federated Learning (CD-FL) is a federated setting where on-devices models are trained, and

TABLE II: Common datasets according to [44], [45], and selected works

Dataset	Type	Records (#)	Attacks (%)	Balanced	Labeled	Classes (#)	Features (#)	Reference
KDD Cup 99	TCPdump data	4,898,431	80.14	✗	✓	5	41	[57]
NSL-KDD	Improved version of KDD99	148,517	53.46	✓	✓	5	41	[58]
AWID	Captured 802.11 packets	37,817,835	2.87	✗	✓	16 / 4	156	[59]
CIDDS-001	NetFlow data	31,287,934	10.34	✗	✓	5	12	[60]
CIDDS-002	NetFlow data	16,161,183	3.48	✗	✓	5	12	[61]
UNSW-NB15	TCPdump data	2,540,044	12.65	✗	✓	10	49	[62]
CICIDS2017	CICFlowMeter	2,830,743	19.70	✗	✓	15	80	[63]

then aggregated to be used by the server. It is especially useful to learn from user data (*e.g.* from smartphones or wearables) while respecting privacy and trust issues [11]. When clients are organizations, in use cases like network security or fraud detection, we use the term Cross-silo Federated Learning (CS-FL) [11].

Most applications are using horizontal federated learning (HFL) [10], which is close to distributed learning. In the case of HFL, the different clients share the same features, but not the same samples. Thus, HFL particularly copes with *ground-truth* issues (Challenge 1) by providing more data for the model to be trained on.

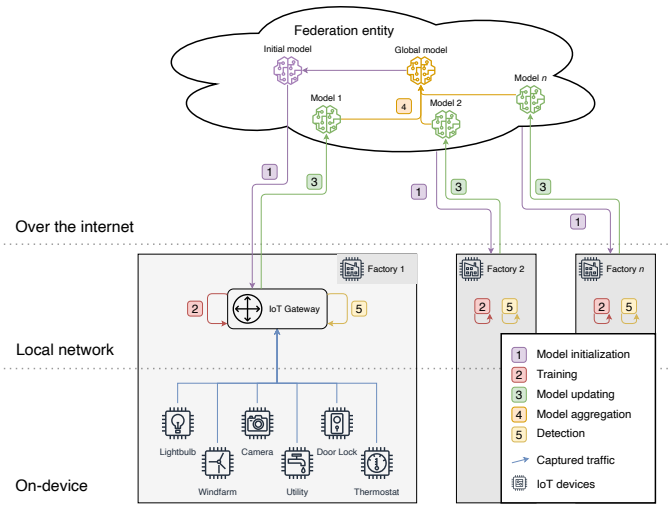


Fig. 2: Example of FIDS

Figure 2 shows an example application of HFL for a collaborative IDS in the context of a smart factory. A network-based gateway captures traffic originated by IoT devices and performs training and detection locally. The federation entity provides an initial model for training and detection. Each model is trained locally, before it is transmitted back to the federation entity. The latter then proceeds with an aggregation of every participant's models. The operation is cyclic as the new aggregated model is disseminated among participants. The process can be repeated until convergence of client models, or until an accuracy threshold is reached. An iteration of this process is called a *round*. More complex aggregation strategies can be used, such as selecting a subset of client for each round, or training specialized models. Section V-C9 details and compares the approaches used by selected works.

The other variants of FL, namely vertical federated learning (VFL) and federated transfer learning (FTL) [10], are less represented in the literature of FIDS. The choice of a FL approach depends on the part of *samples* and *features* that is shared by clients. A sample is an individual entry in a dataset. Features are measurable characteristics of this entry [69]. When storing a dataset as a matrix for processing, each row represents a sample, and each column a feature [10].

HFL is applicable when clients share features but not samples, which is the case in most reviewed works, as denoted in Section V-C8. VFL is the opposite: clients share samples but have access to different feature spaces. An identifier is shared among the samples, so a correlation can be done between the samples of different clients. In FTL, only a subset of both, features and samples, is shared. FTL is often used to transfer the knowledge of a well-trained model to a slightly different use case or context, *e.g.* different networks configurations or device types.

D. Performance metrics

Section V-C12 compares the selected works in terms of evaluation strategies. Therefore, performance metrics and their formula are detailed here. Table III contains definitions for the notations used in the following equations.

TABLE III: Symbols for performance metrics

Symbol	Meaning
TP	True positives
TN	True negatives
FP	False positives
FN	False positives
P	Positive cases
N	Negative cases

ML-based IDSs in the literature are evaluated using common performance metrics [36], [42], [44]. As intrusion detection tasks are a binary classification problem, metrics are obtained by testing the algorithm against a labeled testing dataset, and comparing the output of the model with expected results. Then, the following metrics can be computed:

(1) *Accuracy* represents the proportion of correctly classified items. It is the ability for the system to correctly distinguish abnormal traffic from legitimate one.

$$Accuracy = \frac{TP + TN}{P + N}$$

(2) *Precision*, or positive predictive value (PPV), is the proportion of correct positive cases among all the cases that have been categorized as positive.

$$Precision = \frac{TP}{TP + FP}$$

(3) *Recall*, or true positives rate (TPR) represents the proportion of true positive cases that have been correctly categorized.

$$Recall = \frac{TP}{P} = \frac{TP}{TP + FN}$$

(4) *Specificity*, or true negative rate (TNR), is the proportion of negative cases that has been correctly categorized.

$$Specificity = \frac{TN}{P} = \frac{TN}{TN + FP}$$

(5) *Fallout*, or false positives rate (FPR), represents the proportion of the positive cases that should have been categorized as negative. A high FPR often requires human intervention after the classification task to filter out the false positive.

$$Fallout = \frac{FP}{N} = \frac{FP}{FP + TN}$$

(6) *Miss rate*, or false negative rate (FNR), relates to the proportion of positive cases that have not been categorized as such. In the context of IDSs, it represents an attack that has been missed by the system. Thus, it is a critical metric for this use case.

$$Miss\ rate = \frac{FN}{P} = \frac{FN}{FN + TP}$$

(7) *F1-Score* is the harmonic mean of precision and recall. It is often used to measure ML algorithm, but is also criticized because of the equal importance it gives to both precision and recall [70].

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

(8) *Mathew Correlation Coefficient (MCC)* is an adaptation of the *Phi* (ϕ) coefficient to confusion matrices. While being mathematically identical, the term is often preferred by the ML community. MMC has significant advantages over the other metrics, as it covers all four categories of the confusion matrix (see [71]). Thus, a high score can only be obtained with high TP and TN, and low FP and FN.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

The mentioned formulas can be adapted to multi-class classification problems, *e.g.* attack classification [42]. Other metrics can also be considered, such as algorithm complexity, training and execution costs, or communication overhead (Section V-C12).

V. Survey

This section contains the results of our literature review. First, it synthesizes the analyses into a reference architecture and a taxonomy for FIDSs. Then, it reviews the quantitative analysis used to answer our research questions related to literature (RQs 3.1 and 3.2). Then, it goes over the comparison of selected works to answer RQ 2.1 on the components of FIDSs and their impact on performance.

A. Data synthesis

The quantitative (Section V-B) and qualitative (Section V-C) analyses provide results that are synthesized in a reference architecture, and a taxonomy. The reference architecture presents the components of FIDSs and their interactions, while the taxonomy provides comparison criteria for the selected works.

1) Reference architecture: This section presents the reference architecture synthesized from the selected works, as depicted in Figure 3. It can be divided in three parts:

- The *Managed system* represents the monitored system, *e.g.* IT network, industrial devices, or health-monitoring wearables. As noticed in Section V-C1, collected data can either concern system or environment behavior. The former relates to information generated by the systems, *e.g.* network traces or resource consumption. The latter refers to what the monitored system operates on, *e.g.* health metrics for medical devices of temperature and atmospheric pressure for building management systems.
- The *Security subsystem* is the core of the architecture. It contains all the system's activities, from model training to detection and counter-measures deployment. Depending on the objectives and constraints, this subsystem can either be run locally like [14] or [27], on a dedicated edge-device as in [18], or in the cloud for centralized learning. The subsystem is assumed to run a device that embeds enough computing power to perform real-time anomaly detection against ML models. It is also capable of training its own model based on collected data.
- The *Collaboration subsystem* provides the sharing features of the system, essentially the model aggregation (Section V-C9). It also provides optional training from other sources, like online datasets.

This architecture has similarities with the principles of autonomic systems, as defined by IBM in 2001 [72]. Their architecture is referred to as Monitor-Analyze-Plan-Execute plus Knowledge (MAPE-K). Classic autonomic systems are local, and therefore use a database to provide *knowledge*. In FIDS, FL fills this role in the reference architecture, as the knowledge is being shared among all agents through model aggregation.

2) Taxonomy for FIDS: The taxonomy depicted in Figure 4 summaries the core components and specificities of FIDSs, as extracted from the selected works and existing related taxonomies (Section III-C). Correlations between the taxonomy items and the system's components can be seen in the reference architecture (Figure 3). It also serves as a framework for

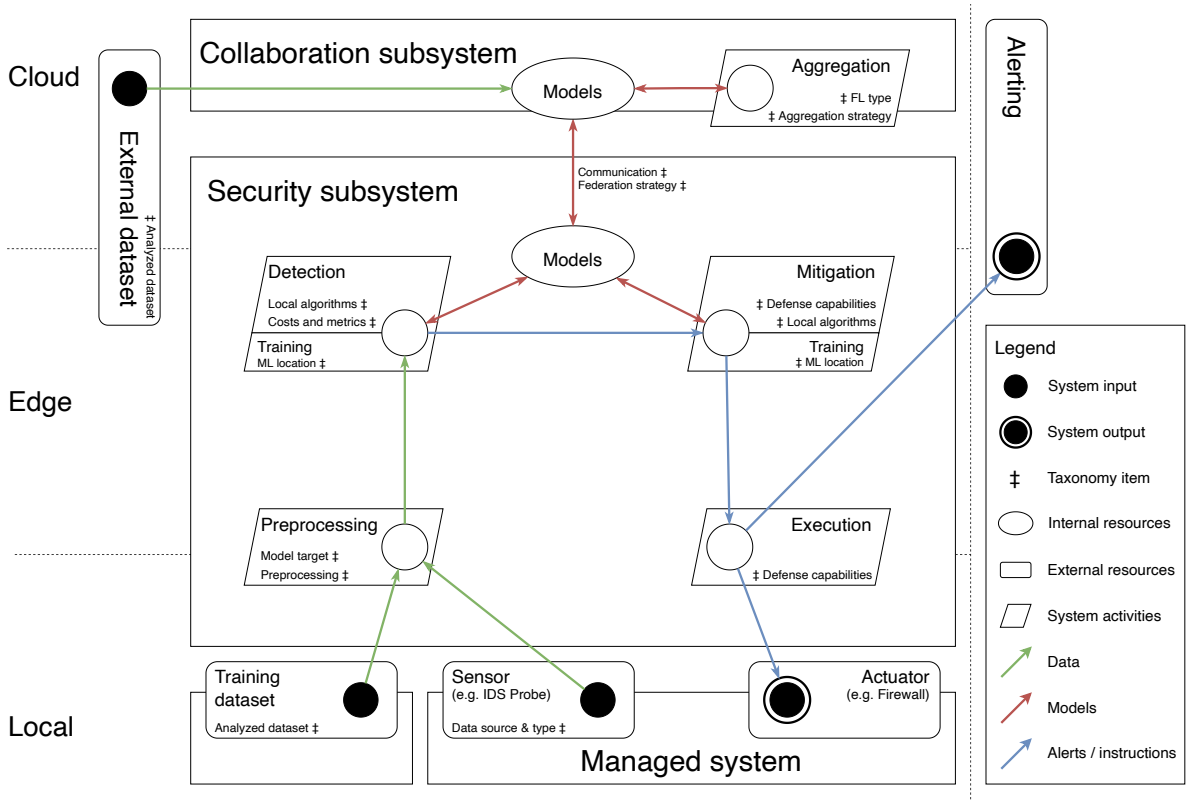


Fig. 3: The reference architecture

the comparisons of the selected works. Each class represents a building block, for which multiple approaches exist depending on use case and constraints.

The proposed taxonomy contains 12 classes describing the selected works that span over five main aspects:

- Two classes cover the topic of *data*: **Data source & type** and **Preprocessing**. It defines what type of data is considered, how it is collected, and the preprocessing strategies that are used.
- *Local operation* is represented by 3 classes: **ML location**, **Local algorithms**, and **Defense capabilities**. It describes the detection and mitigation strategies, how models are built and trained, and where the computing resources are located.
- The *Federation* aspect is covered by 2 classes: **Federation strategy**, and **Communication**. They relate to the communication between the agents and the server, and how data sharing is organized.
- *Aggregation* is also covered by 3 classes: **FL type**, **Aggregation strategy**, and **Model target**. It describes the type of FL used, how the models are fused, in accordance with the objectives of the system.
- Finally, 2 classes define the *Experimentation* topic: **Analyzed dataset** and **Costs and metrics**. This meta-category does not relate to the proposed solution, but to how the experiments are performed.

B. Quantitative analysis

This section analyses several indicators of the representation of FIDSs in the academic literature: the evolution of the topic, the active groups, and the venues in which the contributions are made. With the study of active research groups and relevant venues, this section provides new sources of papers for snowballing (see Section III).

1) Evolution of the topic: The topic of IDS started to gain interest in the early 2000' as depicted in Figure 5a. After a few years, the topic regained interest in 2015, with an increase of research on IoT and Industrial Internet of Things (IIoT) [44], [74], and other specific use cases. With the introduction of FL by McMahan *et al.* [7], the community started to apply FL approaches to IDS around the years 2018–2019 (Section V-C). Figures 5a and 5b have been generated using *Topics* feature of Microsoft Academic, which allow to quickly gather statistics and research papers about a given topic. The following topic queries have been used, matching Topics (b) and (c) defined in Section IV:

(a) `intrusion detection system`;
(b) `federated learning`.

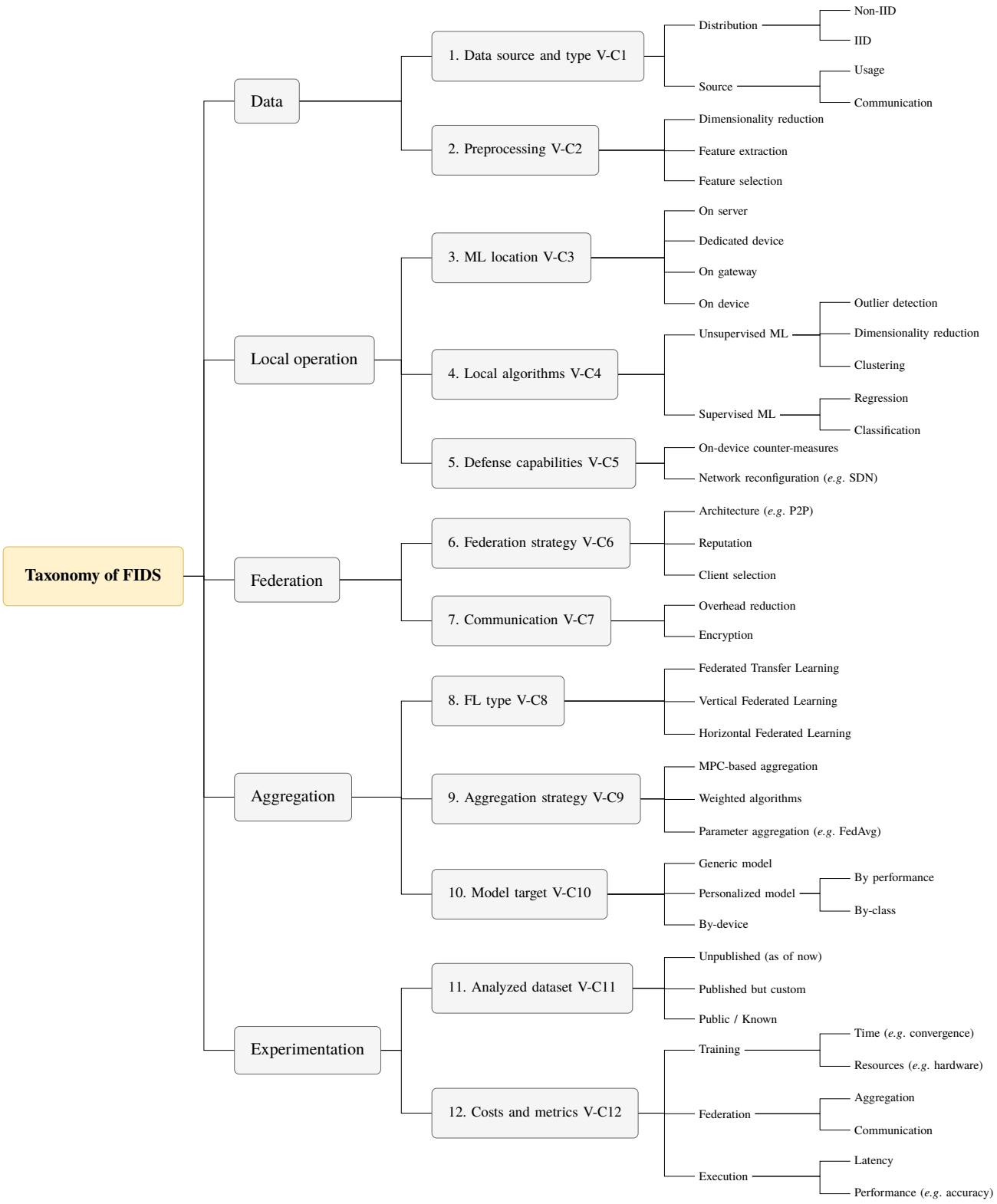
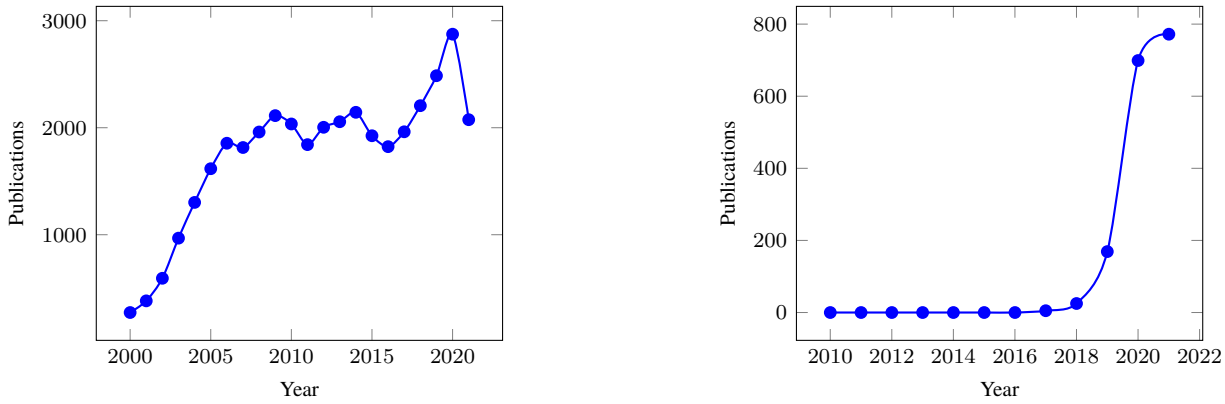


Fig. 4: Proposed taxonomy for FIDS



(a) Intrusion Detection System (IDS)—Query (a)

(b) Federated Learning (FL)—Query (b)

Fig. 5: Evolution of the topics using Queries (a) and (b), according to Microsoft Academic [73], until 2021-10-08

Recent works on FL focus on its security and privacy-preserving aspects [8], [38], [48]. Techniques like homomorphic encryption has been introduced as early as 2017 [75], and have been extensively reviewed since. More recently, other privacy-preserving techniques have been applied to FL, such as multi-party computation (MPC) in FLGUARD [76] or differential privacy in [77]. This tendency towards algorithm security is also represented in FIDSs. For instance, Li, Wu, *et al.* [18] use homomorphic encryption to provide a secure and privacy-preserving aggregation of models. Aside from security, variations of HFL started to appear in 2021, such as segmented FL in [23], as standard HFL has significantly been studied for FIDS.

Finally, literature reviews published in 2021 [33]–[35] show the interest of the community for the study of FIDSs. These also show the need for synthesis and structuring of research in this area.

2) Relevant venues: Venues are very diverse in the selected works. Different IEEE conferences (ranked B and higher [79]) are represented, such as IEEE International Conference on Distributed Computing Systems (ICDCS) [12], IEEE International Conference on Computer Communications and Networks (ICCCN) [15], and International Conference on Network and Service Management (CNSM) [14]. Conferences are often favored for the presentation of technical results, and are therefore well represented in the reviewed papers (11), alongside with journals (10), and books (1). Only three venues are represented twice in our selection: *IEEE Internet of Things Journal* [20], [30], *IEEE Access* [13], [28], and the *IEEE BigDataSE* conference [17], [21]. Figure 6 shows the relevant venues in the literature with their types and the number of concerned papers.

The diversity of the venues for this topic can be explained by the number of use cases where FL–based intrusion detection can be applied. The *IEEE Internet of Things Journal* for instance targets the domain of IoT and connected devices, whereas the *IEEE Transactions on Vehicular Technology* is more focused on connected vehicles. Nevertheless, both contain papers on FIDS. As publication choices are currently motivated by use cases, research venues are not yet representative of FIDSs as a topic, but rather as a technique with diverse applications.

3) Most active groups: Since they introduced the topic of FL in 2016, the team at Google Research has been a big influence for the research community [7], [65], [67], [80], [81]. They mostly work on the primitives behind FL, such as model aggregation with the FedAvg algorithm [7]. The team of TU Darmstadt (Germany) is contributing to the field with DÏoT [12], [82] and an analysis of FL poisoning attacks [8]. More recently, they produced FLGUARD [76], in collaboration with Google and the Aalto University (Finland), which makes them the most represented team in the field. The University of Tokyo is represented by three recent works [23], [31], [32] in the last two years on FL and segmented FL for network intrusion detection.

Other groups are represented in the selected works once. China is a major contributor, with seven different universities contributing to the field [19]–[21], [29]. Figure 7 shows the most active groups and their publications, with their topic focus expressed with color.

C. Qualitative analysis

This section reviews the selected literature. The selected works are detailed, and compared using the taxonomy. Table IV summarizes the information and helps identify differences between the works. It gives partial answers to research questions about the components of FIDSs and how to measure their impact on performance (RQs 2.1 and 2.2), while Section V-C9 replies to RQ 1.2 about federation techniques.

1) Data source and type: Depending on the use case, there are two main approaches. The first one is the one used by [15], [20], where the model is made on the sensors' values, which is analogous to the operation of Host-based Intrusion Detection System (HIDS). Since [20] targets medical devices, values include hearth rate, oxygen saturation, among others. The opposite strategy function at a higher level of abstraction, independent of the values. The analysis is then performed

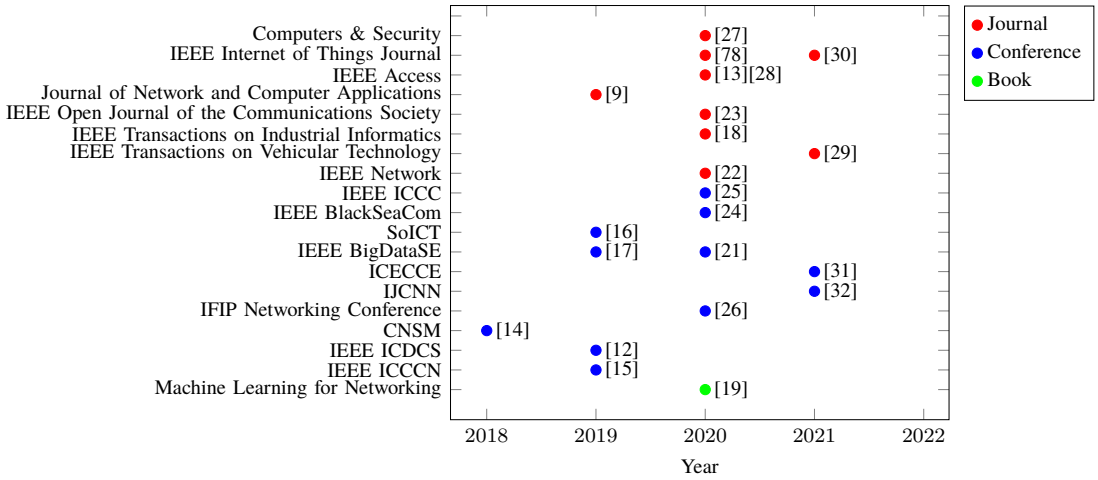


Fig. 6: Relevant venues

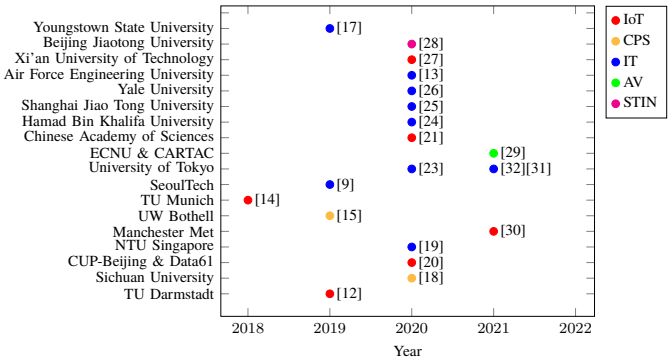


Fig. 7: Research groups

on network traffic, as for Network-based Intrusion Detection System (NIDS). Most papers [9], [12], [13], [18], [19], [22], [23], [27], [30] use similar network features, such as source and destination, local and remote ports, TCP flags, protocol, and packet length. The authors of [26] also target network features but at packet-level, all translated to 1D vectors: IP addresses, layer-4 protocol, ports, and IP packet length as a 120-bit input vector. Li, Zhou, *et al.* [28] also explore network-related features in their use case of satellite communications.

These values can be completed with preprocessing (see Section V-C2) to extract other features from the raw data. In both [14] and [12], the periodicity of packets is analyzed. This is important for volumetry attack detection notably. Furthermore, both work target IoT devices, which have a sporadic, but periodic and predicable traffic. Thus, anomaly in the packet-sequence, or in the inter-arrival time might indicate an attack. While following a similar approach, FTL allows the authors of [21] to address different features in each client's dataset.

The research led by Pahl *et al.* [14] differs from the other on the source for the data. They use a middleware called Virtual State Layer (VSL) to associate traffic with a device class (*light*, *router*, *sensor*, …), thus allowing the training of per-class models with high accuracy. However, many OT solutions do not provide such metadata. Training per-class models requires then a prior classification step, like in [12].

Additionally, even when considering the same data type, use cases introduce significant differences in the available features. For instance, two systems targeting the communication between devices may encounter different protocols, services, and even communication support. Among the selected works, four use cases are considered, here sorted by representation:

- Information Technology (IT);
- Internet of Things (IoT);
- Cyber-Physical System (CPS);
- Autonomous Vehicles (AV).

The work of Liu, Zhang, *et al.* [29] is the only representative of the Autonomous Vehicles (AV) use case, although they do not use an according dataset. In fact, they train their model on network traffic, with similar features to [9], [19], [21], [22]. With also similar features, Li, Zhou, *et al.* [28] apply FIDS to the very specific use case of Satellite-Terrestrial Integrated Network (STIN).

Finally, [13], and partly [27], address data distribution, especially knowing whether data are Independent and Identically Distributed (IID). A non-IID data distribution can negatively impact training performance [10]. However, most real-world scenarios generate non-IID data, which is a major hurdle for algorithm that require to be trained on live data with non-supervised approaches.

2) Preprocessing: The source data can be manipulated to extract new feature or reduce dimensionality through preprocessing. Three main non-exclusive approaches are distinguishable in the selected works: feature extraction, feature embedding, and feature selection:

- *Feature extraction* refers to the computation of numerical characteristics after the data collection; *e.g.* inter-arrival time (IAT) or number of packets per device in the context of traffic monitoring. This approach is taken by Pahl *et al.* [14] and Nguyen, Marchal, *et al.* [12] as they proceed

TABLE IV: Comparative overview of selected works—approach and objectives (1/2)

	Ref	Internet of Things	Information Technologies	Cyber Physical Systems	Autonomous Vehicles	Satellite-terrestrial networks	Horizontal FL	Vertical FL	Federated Transfer Learning	Federated MTL	Federated Mimic Learning	Online learning	Supervised	Semi-supervised	Unsupervised	Personalized models	Network-based	Usage-based	Training location	Data type	Strengths
		Use case					FL type						Training				Approach				
2018	Pahl *et al.* [14]	●	○	○	○	○	●	○	○	○	○		●	○	○	●	●	●	○	Device	Abstracted network traffic (middleware)	relatively lightweight, online, no labels
2019	Rathore *et al.* [9]	○	●	○	○	○	●	○	○	○	○		○	●	○	○	●	●	○	Edge-controller (SDN)	Network traffic (SDN)	offers mitigation, decentralized
2019	Nguyen, Marchal, *et al.* [12]	●	○	○	○	○	●	○	○	○	○		●	○	○	●	●	●	○	Gateway	IoT network traffic (TCPdump)	online, offers per-class models, no labels
2019	Zhao *et al.* [16]	○	●	○	○	○	○	○	○	●	○		○	●	○	○	●	●	○	Gateway	Encrypted network traffic (CICFlowMeter)	versatile (multi-task)
2019	Schneble *et al.* [15]	○	○	●	○	○	●	○	○	○	○		●	○	○	●	○	○	●	Gateway	Healthcare sensor values	high adaptability, no labels
2019	Cetin *et al.* [17]	○	●	○	○	○	●	○	○	○	○		○	●	○	○	○	●	○	Gateway	Network traffic (WIFI)	–
2020	Zhang, Lu, *et al.* [20]	●	○	○	○	○	●	○	○	○	○		○	●	○	○	○	○	●	Gateway	Air conditioner sensor values	offers traceability (blockchain)
2020	Li, Wu, *et al.* [18]	○	○	●	○	○	●	○	○	○	○		○	●	○	○	○	●	○	Gateway	MODBUS traffic	confidentiality (encryption)
2020	Rahman *et al.* [22]	○	●	○	○	○	●	○	○	○	○		○	●	○	○	○	●	○	Device	IoT network traffic (TCPdump)	–
2020	Chen, Zhang, *et al.* [19]	○	●	○	○	○	●	○	○	○	○		○	○	○	●	○	●	○	Gateway	IoT network traffic (TCPdump)	no labels
2020	Sun, Ochiai, *et al.* [23]	○	●	○	○	○	●	○	○	○	○		○	●	○	○	●	●	○	Gateway	Network traffic (PCAP)	segmented (performance-based models)
2020	Fan *et al.* [21]	●	○	○	○	○	○	○	●	○	○		○	●	○	○	●	●	○	Gateway (MEC)	IoT network traffic (TCPdump, CICFlowMeter)	knowledge transfer between public and private datasets
2020	Al-Marri *et al.* [24]	○	●	○	○	○	○	○	○	○	●		○	●	○	○	○	●	○	Gateway	Network traffic (TCPdump)	enhanced privacy (mimic learning)
2020	Kim, Cai, *et al.* [25]	○	●	○	○	○	●	○	○	○	○		○	●	○	○	○	●	○	Gateway	Network traffic (TCPdump)	–
2020	Qin, Poularakis, *et al.* [26]	○	●	○	○	○	●	○	○	○	○		○	●	○	○	●	●	○	Gateway (SDN)	Network traffic (SDN)	very lightweight, line-speed classification, P4 language compatible
2020	Chen, Lv, *et al.* [13]	○	●	○	○	○	●	○	○	○	○		○	●	○	○	○	●	○	Gateway	Network traffic (CICFlowMeter)	robust to poisoning, scalable
2020	Hei *et al.* [27]	○	●	○	○	○	●	○	○	○	○		●	○	●	○	○	●	○	Device	Network traffic (TCPdump)	online, offers traceability (blockchain)
2020	Li, Zhou, *et al.* [28]	○	●	○	○	●	●	○	○	○	○		○	●	○	○	○	●	○	Gateway	Network traffic (PCAP, CICFlowMeter, Argus)	relatively lightweight, confidentiality (encryption)
2021	Popoola *et al.* [30]	●	○	○	○	○	●	○	○	○	○		○	●	○	○	○	●	○	Gateway	IoT network traffic (TCPdump, Argus)	zero-days detection
2021	Qin and Kondo [31]	○	●	○	○	○	●	○	○	○	○		○	●	○	○	●	●	○	Device	Network traffic (TCPdump)	relatively lightweight
2021	Liu, Zhang, *et al.* [29]	○	○	○	●	○	●	○	○	○	○		○	●	○	○	○	●	○	Device	Network traffic (TCPdump)	decentralized
2021	Sun, Esaki, *et al.* [32]	○	●	○	○	○	●	○	○	○	○		○	●	○	○	●	●	○	Gateway	Network traffic (PCAP)	segmented (performance-based models)

with periodicity mining. Since they only process binary features, Qin, Poularakis, *et al.* [26] extract numerical features, and convert them to 1D vectors.

- *Feature embedding* or *dimensionality reduction* is used for algorithms that do not deal efficiently with high-dimensional vectors. To that end, they use data dimensionality reduction techniques, such as autoencoders [19] or Principal Component Analysis (PCA) [25].
- *Feature selection* relates to the automated selection of relevant features, before learning. The authors of [31] use a greedy feature selection algorithm based on accuracy. Logistic regression-based selection [24] can also be used to eliminate features with a recursive algorithm.

The other works [9], [15], [18], [78] do not emphasize on their feature selection strategy. Moreover, some papers [15], [16], [18] use datasets that contains computed features (V-C11). For experiments on live prototypes, feature computation is required.

Depending on the use case, additional features after *feature selection* or *extraction* may vary. Network analysis often relies on basic features, such as addresses and ports for source and destination, protocol, data type, packet length, and timestamp. However, these characteristics can also vary regarding their provenance: network capture [57], [58] or abstracted communications [14]. Extracted features are very common, such as inter-packet time, bytes sent per host, or bytes per packets [42], [44].

Usage-based analysis, on the other hand, is entirely dependent on the monitored device. Schneble *et al.* [15] monitor health-related features, like arterial blood pressure or the raw ECG signals. The authors of [20] focus on air conditioners, and therefore measure related information such as water or air

temperature.

3) ML location: As explained in Section V-A2, the location of ML algorithm in a system influences the architecture. The proposed taxonomy (4) considers three types of locations: on-device, on-gateway, and on-server. However, a large majority of the literature concerns either on-device training, or uses a dedicated device acting as a gateway. Most selected works use a dedicated device to perform the analysis, while the others assume the devices can support their own processing. The on-server processing is not represented here, since it does not suit the definition of FL. Some hybrid approaches are also represented, with multi-stage aggregation [29].

The device types and architectural choices vary, depending on the use case. Zhang, Lu, *et al.* [20] focus on a medical use case where the analyzed data is composed solely of sensor outputs (Section V-C1). Connected sensors are typically lightweight devices unable to process data. Thus, they require a gateway to be usable. Other works [15], [16], [18], [19], [24], [25], [28], [30] rely on gateways because they are more suitable for traffic analysis. It allows to capture all communications, even if the devices are connected with different supports (*e.g.* IEEE 802.3 *versus* IEEE 802.11). Gateway-based processing can also be motivated by the architecture of the monitored system. For instance, the authors of [21] reuse the existing infrastructure of 5G by exploiting Mobile Edge-Computing (MEC) gateways to capture traffic and perform analysis for a 5G IoT use case.

While also using gateways, Rathore *et al.* [9] differ by relying exclusively on Software-defined networking (SDN) switches to analyze the traffic and provide their counter-measures. In this case, learning and detection does not happen on the gateway itself (the SDN switch), but on an SDN controller in charge of managing a fleet of switches. While this approach employs an intermediary location for model training and decision-making, it cannot be considered as cloud based as the models are aggregated in a cloud server afterward. While they have a similar architecture, Qin, Poularakis, *et al.* [26] propose the opposite approach. The algorithm is deployed on the SDN switch, allowing faster response time by examining the traffic at packet-level. Their Binarized Neural Network (BNN) detection algorithm (Section V-C4) enables local detection close to real-time, even with high packet throughput, such as during a Denial of Service (DoS) attack. The authors test both approaches, and find similar performance in terms of accuracy, precision, recall, and F1-score.

Pahl *et al.* [14] take another approach and assume the IoT devices to be powerful enough to run their own analysis and training. Their design cuts out the gateway by using a middleware which allows peer-to-peer (P2P) communication between the agents, thus removing the need for a gateway. An *IoT Microservice Store* provides federation and model aggregation, introducing different hierarchies of collaboration. This assumption is also taken in [22], [27] and [31]. Liu, Zhang, *et al.* [29] also train their models "on-device" as their use case target AV. Such vehicles often carry consequent processing abilities for environment recognition alone, and are thus assumed to be able to perform ML training.

4) Local algorithms: Most IDSs fall into one of the following categories: anomaly-based, signature-based, or specification-based. Hybrid systems are also considered, but to the best of our knowledge, no example exist in the literature of FL-based detection systems. Apart from preprocessing [83], ML-based detection systems mostly rely on the detection of anomalies, and thus exclude signature- and specification-based detection. As introduced in Section IV, depending on the presence of labels, three approaches coexist:

(1) Supervised learning transforms the anomaly detection into a binary classification problem. It requires a balanced labeled dataset for the training to be effective. However, it is harder to deploy in real condition as local training data are rarely labeled, and models can be skewed by unbalanced class distribution [34]. This approach is yet chosen by most of the selected works (16 out of 22).
(2) Unsupervised learning is suitable for unlabeled data. In the specific use case of IDS, it assumes that (i) benign traffic is much more frequent that anomalies in the testing set [84]; (ii) abnormal packets are statistically different from normal ones. Unsupervised learning is used by 5 of the selected works.
(3) Semi-supervised learning is a hybrid approach where only a small part of the training data is labeled. It can be used to bootstrap a detection model by using a public labeled dataset, but then training it on local captured data. This newer paradigm is almost not represented in the selection, but more recent works—published after the submission of this survey—focus on semi-supervised learning [85], [86].

Only four of the selected works adopt online learning [12], [14], [15], [27]. Online learning refers to the ability to train a model continuously as data arrives. It provides great adaptability and allows the algorithm to follow the evolution of the monitored system. All online work in the selection use either unsupervised or semi-supervised approaches, as continuously feeding labeled data is impracticable. The opposite approach, offline learning, refers to a one-shot training on a defined training set. Between the two, re-training enables updating the models to fit more recent data, but this is not particularly addressed in the selected work.

Further differences emerge between the chosen algorithms. There is a strong representation of solutions based on Neural Network (NN) (21 out of 22), as shown in Table V. However, few use the same approach. The selected components (*layers*) differ. Nguyen, Marchal, *et al.* [12] leverage the capabilities Recurrent Neural Network (RNN) to detect unusual packets, given a sequence of traffic. Gated Recurrent Unit (GRU) are a type of RNN known to be very efficient in terms of resource consumption. They allow to keep a *history* of the precedent processed values, which is useful for context keeping or pattern recognition. In this case, the packet history is used to detect deviant traffic, and raise an alert. The authors of [13] also used a GRU–based NN, but replaced the `Softmax` function of the

last layer by a Support Vector Machine (SVM) one to improve performance, as it is stated to improve with linear functions.

Other bricks can be used to improve the processing. Li, Wu, *et al.* [18] add a Convolutional Neural Network (CNN) and a Multilayer Perceptron (MLP) to improve their model performance, which is said to surpass both [12], [15]. CNNs excel at analyzing complex pattern without performance issue. Fan *et al.* [21] also implement CNN in the shared layers of their FTL approach, the last layers being fully-connected ones. The authors of [28] also experiment on CNN models, with one, two, or three hidden layers, but observe decreasing performance as the number of layers increases.

While they can be used together, CNN are often used as a replacement for standard MLP. The authors of [15] use a single hidden-layer MLP to classify the measurements as normal or abnormal. Since they use medical measurements as input (Section V-C1), the MLP is trained to recognize out-of-range values or correlation issues (two linked values are supposed to evolve in the same way). MLP is also used in [22], [25], [27], [29]. While they provide significantly lower performance, their use in FL research can be motivated by their potential for easy aggregation. Moreover, advances in FL do not rely on the local algorithm, but more on federation strategies.

In [20], the learning is performed with a NN with two hidden layers. The last layer is a `Softmax` function which returns a probability of being in a class (normal or abnormal), which is applicable for most classification-based approaches [18], [21]–[24], [30], [32]. Rathore *et al.* [9] also rely on Deep Learning (DL) and NN, with an unspecified number of hidden layers. In their work, the authors of [19] propose a combination of an autoencoder for dimensionality reduction, and a Gaussian Mixture Model (GMM) for classification. The entire process is autonomous and does not require labeled data. An autoencoder is also used by [31], in combination with Extreme Learning Machines (ELMs). ELMs are feedforward NNs, just as MLPs, where the weights of the neurons are set once, and never updated. This leads to good generalization capabilities and fast training, but lower performance. While using DL, Zhao *et al.* [16] also differ from the others by implementing Multi-Task Learning (MTL), where different models are trained as declination of one common base model. The model is otherwise simpler, as it is mostly made of activation and dropout layers stacked.

The approach of Qin, Poularakis, *et al.* [26] differs in the type of NN model used. To achieve *line-speed* packet processing, their model needs to be executed on data-plain SDN devices, which only support a subset of operations when compared to regular network devices. To cope with that limitation, the authors deploy a BNN algorithm. BNNs are a category of NNs with only binary weights, activation functions, and the according bitwise operations, allowing fast execution and efficient memory consumption. These characteristics make them suitable for low-level network detection.

Pahl *et al.* [14] are the only ones not using NNs, but simpler clustering algorithm *k*-means, which require fewer data to be trained. Before that, and to optimize the speed of the *k*-means algorithm, grid-based clustering is used to identify clusters quicker. The clustering is used for periodicity-mining (Section V-C1), on which the approach is based. Detection is based on both the difference between current and previous communications, and the likelihood of the message (depending on how often two devices communicate together). Moreover, while non-DL machine learning is under-represented in the literature of FIDS, the authors of [14] are not the only ones to experiment on non-DL ML. Hei *et al.* [27] review other alternatives like Decision Tree (DT), Random Forest (RF), or SVM, but the MLP obtains better performance overall. Schneble *et al.* [15] train K-nearest neighbors (KNN), DT, Stochastic Gradient Descent (SGD), and SVM alongside their Neural Network.

5) Defense capabilities: Defense strategies are barely covered in the selected works. Only one paper provides actionable counter-measures. The work of Rathore *et al.* [9] builds upon SDN, which allows the controller to modify the network architecture in case of an attack. The proposed solution is tailored for DoS or flooding attacks, and therefore only needs to block the responsible traffic flow.

FIDSs could also provide remediation capabilities, providing automated resilience of a monitored system [98]. To the best of our knowledge, there is no such work in the literature. However, multiple works have been proposed to provide self-healing behaviors to information systems [99], [100]. Such functionalities could be considered to enhance FIDS capabilities.

6) Federation strategy: FIDS literature shows how the number of clients can impact performance. In [12], while the FPR decreases to zero with the augmentation of clients, the TPR also decreases slightly—from 95.43% to 94.07% by going from 5 to 15 clients. Other works observe only positive results, with a small accuracy increase (0.002% from 1 to 8 clients) in [15] for instance, while Li, Wu, *et al.* [18] measured very stable results when going from 3 to 10 clients.

Consequently, massive FL applications often implement a client-selection algorithm which only train a subset of participant at each round, thus reducing the computing load and bandwidth consumption. This selection can even be done dynamically on performance metrics [78], but has not been found in the literature on FIDS.

On an architectural perspective, most of the selected work follow a client-server model, where clients train models locally and a centralized server proceeds with model aggregation. While relatively easy to deploy, such approach has caveats, such as the necessity of trusting the central server, or the SPoF in the aggregation process [47].

Therefore, the authors of [9], [20], [27], [29] rely on decentralization in their design. Zhang, Lu, *et al.* [20] justifies its use of blockchain as a way to ensure integrity for the detected anomaly, in a failure-detection context. On the other hand, the authors of [9] use the blockchain as a decentralized storage and aggregation service to improve resiliency by removing the SPoF. Liu, Zhang, *et al.* [29] rely on distributed ledgers for

TABLE V: Comparative overview of selected works—algorithms and performance (2/2)

	Ref	Local Algorithm	Federation Algorithm	Accuracy	Precision	Recall	Fall-out	F-Score	K [a]	Dataset
2018	Pahl *et al.* [14]	BIRCH K-means	Parameter addition	0.9900	–	0.9600	0.0020	–	7	Generated
2019	Rathore *et al.* [9]	ANN	Vector concatenation	‡ 0.9100	‡ 0.9100	‡ 0.9100	–	‡ 0.9100	15	NSL-KDD [58]
2019	Schneble *et al.* [15]	MLP	Weight and biases average	0.9930	–	–	–	–	64	MIMIC [87]
2019	Nguyen, Marchal, *et al.* [12]	GRU	FedAvg	–	–	0.9543	0	–	15	Generated
2019	Zhao *et al.* [16]	FC (shared layers) → FC	Weight and biases average	∗ 0.9797	∗ 0.9634	∗ 0.9681	–	–	–	CICIDS2017 [63] ISCXVPN2016 [88] ISCXTor2016 [89]
2019	Cetin *et al.* [17]	SAE	FedAvg	–	–	–	–	–	933	AWID [59]
2020	Li, Wu, *et al.* [18]	CNN-GRU → MLP	Homomorphic parameter addition	0.9920	0.9885	0.9745	–	0.9813	7	CPS dataset [90]
2020	Chen, Zhang, *et al.* [19]	DAGMM	Parameter addition	–	0.7447	0.9803	–	‡ 0.8700	2 [b]	KDD 99 [57]
2020	Zhang, Lu, *et al.* [20]	ANN	CDW_FedAvg	∗‡ 0.8900	∗‡ 0.8600	∗‡ 0.9450	–	∗‡ 0.8500	4	Generated
2020	Fan *et al.* [21]	CNN	Parameter aggregation	∗ 0.9100	–	∗‡ 0.9350	∗‡ 0.0020	–	4	CICIDS2017 [63] NSL-KDD [58] Generated
2020	Rahman *et al.* [22]	ANN	FedAvg	∗ 0.7731	–	–	–	–	4	NSL-KDD [58]
2020	Sun, Ochiai, *et al.* [23]	CNN	Parameter aggregation	∗ 0.8710	–	–	–	–	20	LAN-Security Monitoring Project [91]
2020	Al-Marri *et al.* [24]	ANN	FedAvg	0.9812	∗ 0.9900	∗ 0.9900	∗ 0.1320	∗ 0.9900	10	NSL-KDD [58]
2020	Kim, Cai, *et al.* [25]	MLP	FedAvg	0.9712	–	–	–	–	4	NSL-KDD [58]
2020	Qin, Poularakis, *et al.* [26]	BNN	SignSGD	∗ 0.9640	∗ 0.9555	∗ 0.8645	–	∗ 0.9055	8	CICIDS2017 [63] ISCX Botnet 2014 [92]
2020	Chen, Lv, *et al.* [13]	GRU-SVM	FedAGRU	∗ 0.9905	–	–	∗ 0.0108	∗ 0.9762	20	CICIDS2017 [63] KDD 99 [57] WSN-DS [93]
2020	Hei *et al.* [27]	MLP	FedAvg	∗‡ 0.8950	∗‡ 0.9750	∗‡ 0.8775	–	∗‡ 0.9225	3	DARPA 1999 [94]
2020	Li, Zhou, *et al.* [28]	CNN	Homomorphic parameter addition	∗ 0.8100			∗ 0.1900		4	Generated
2021	Liu, Zhang, *et al.* [29]	MLP	Parameter aggregation	‡ 0.9600	0.9400	0.9500	–	–	6	KDD 99 [57]
2021	Popoola *et al.* [30]	ANN	FedAvg	∗ 0.9939	∗ 0.9819	∗ 0.9676	–	∗ 0.9728	5	Bot-IoT [95] N-BaIoT [96]
2021	Qin and Kondo [31]	ONLAD [97] (ELM + AE)	FedAvg	0.7040	–	–	–	–	8	NSL-KDD [58]
2021	Sun, Esaki, *et al.* [32]	CNN	Parameter aggregation	–	–	–	–	∗ 0.8930	20	LAN-Security Monitoring Project [91]
						Metrics				

∗ Value is an average of those provided by the authors.
‡ Value is read from a graph in the article, and may vary a few from the exact value.

[a] K is the highest number of client considered in the experiments.
[b] Chen, Zhang, *et al.* [19] measure how one client performs, by training one other.

traceability and integrity, but also to support the aggregation between roadside units (RSUs), in a decentralized manner. Their aggregation process has two stages. Firstly, in P2P between the vehicles themselves, and secondly between the RSUs—which connect vehicles to the rest of the world in the Vehicle-to-Everything (V2X) paradigm. Finally, Hei *et al.* [27] use the Hyperledger Fabric [101] to provide integrity and redundancy.

7) Communication: FL relies on communication to share models between participants, which can be compelling in constrained environments [26]. Therefore, some selected works try to reduce the communication overhead generated by their solution. Pahl *et al.* [14] show that their BIRCH cluster approach reduces packet size (from 169 bytes for the average packet to 96 per model), but also communication frequency by sending only one packet per minute.

The authors of [15], [20] compare the communication used by their system in model sharing, and compare it with the dataset size, which would require to be transferred in non-FL settings. While their results show that the relevance of FL to limit communication usage can be questioned in small datasets, its strength is undeniable with standard use cases—above 10^5 bytes according to [20]. While the communication overhead is one of the advantages of FL over centralized ML, it is not often considered in the literature.

Communicating the model parameters can also impact its confidentiality. The authors of [18] and [28] use homomorphic encryption to aggregate the parameters without the server knowing the generated model. The Paillier cryptosystem supports addition [102], which is performed on the server, before the result is disseminated back to the clients. Each client can then decrypt the generated model, and devise the parameters by the number of participants to obtain the averaged biases and weights.

8) FL type: As introduced in Section IV-B, most FL implementations use HFL, 18 out of 22. VFL does not appear in the literature, and is yet to be applied to the use case of FIDS. As VFL requires having the same samples but different features, it is not applicable to collaborative IDSs. Having the same samples would mean that the different participants monitor the same devices, just using different features, which does not follow the motivations of this work. Nevertheless, VFL might be relevant for correlation purposes in a local architecture.

On the other hand, some papers show that FTL can be used to train models in different but related contexts. For instance, a

model trained on the periodicity of specific devices as in [12], [14] would not perform well against devices with behaviors that are too different. However, with FTL, one could quickly train a local model specific to his devices, using the knowledge acquired previously by others, as in [21]. Another application of this concept is used by [16] with MTL, where a same model is trained simultaneously for multiple tasks. Like in FTL, the model is retrained after the sharing to have personalized behavior.

Al-Marri *et al.* [24] implement Federated Mimic Learning (FML) to improve data privacy. Mimic learning is a technique that use two models and two datasets to train and share information afterward. *Teacher* model is trained on the real and sensitive data, and used to label a public dataset. *Student* model is then trained on the newly labeled public dataset, and shared with other participants after that.

9) Aggregation strategy: The aggregation strategy is at the core of FL. In 2016, Google proposed `FedSGD` [7] along with the concept of FL. `FedSGD` is a weighted average of the local gradient. SGD is commonly used in NNs as an optimization function; in fact, most research in the past was about adapting models, so they can be efficiently solved with SGD [7], [103]. Thus, gradient aggregation is stated to be the most logic choice. However, aggregating the gradient at each epoch is costly in terms of bandwidth consumption and computing power. Therefore, the authors introduce `FedAvg`, with the aggregation of model parameters instead of gradient. This let each client training the model locally and only sharing the updates after multiple epochs. This approach is the base of most implementations going forward.

Multiple articles [12], [24], [25], [30], [31] use directly `FedAvg` in their work, just as [22], their contributions being centered on the anomaly detection. In [20], a variant of `FedAvg`, called `CDW_FedAvg` is used. To enhance the performance of their system, the authors weight the average according to the distance between the positive and negative classes of the client using centroid distance. The Centroid Distance Weighted (CDW) average allows to reduce the impact of the heterogeneity in the IoT. The heterogeneity issue is a big motivation for finding alternate aggregation methods. While not focusing on intrusion detection, Sun, Lei, *et al.* [104] propose an asynchronous FL strategy based on node clustering and a deep Q-network (DQN) Reinforcement Learning (RL) algorithm that identifies the best aggregation frequency in real time.

In [13], the authors propose another `FedAvg`–based approach which adds attention mechanism to decrease the communication cost by selecting clients. The clients measure the performance of their model, and update new weights accordingly, the larger the weight the better. When aggregating the client parameters, the server uses the weights as a representation of the client's importance, and thus the quality of its model. In fact, the number of considered clients has been shown to alter the results significantly (Section V-C6).

Other articles average the weights and bias in the NN matrices [9], [15], [19], while not mentioning `FedAvg` explicitly. Thus, the obtained matrix is used to define the hidden layer (often the last if there are more than one) of the model. Li, Zhou, *et al.* [28] propose further optimizations of the FL algorithm to suit their use case of satellite-terrestrial communications. The aggregation of weights and biases is also used by [21], though not for HFL but FTL. The lower layers of DL models learn the more generic features [105], thus allowing the sharing of only the relevant information, and letting the last layers learn personalized features only applicable to the local network.

Because they use BNNs with only binary values, the authors of [26] cannot simply average the model parameters. While the last layer of the BNN could be converted to numerical values to be aggregated more easily, the authors prefer the binary approach SignSGD [106]. This aggregation algorithm relies on majority voting to estimate the best weights for the layers. While their system performs well, the authors point out that updates that do not change the sign of the weights represent a waste of resources, since only two values are possible, $+1$ or -1.

Pahl *et al.* [14] use balanced iterative reducing and clustering using hierarchies (BIRCH) clusters, which have the particularity of being easily aggregated by simply adding the features of multiple clusters together. The model fusion is performed in a *microservice (μS)-store* that distributes and monitors the stored models. Timestamps are also saved to detect the *staleness* of the clusters.

10) Model target: Training an efficient generic model can be difficult, and yield unsatisfying results. As highlighted in [12], anomaly detection systems suffer from higher false positive rate, and lower sensitivity when monitoring different behavior at the same time. To solve this problem, Nguyen, Marchal, *et al.* [12] add an autonomous classification module [82] that allow them to classify devices first, and then train per-class models afterward. The authors of [9] also train specific models with their *traffic classifier*, but do not specify how.

This classification problem is not a security-only issue, and standards have been proposed for devices to advertise information; Manufacturer Usage Description (MUD) [107] for instance allows devices to signal to the network what type of functionalities and authorizations they require to function properly. While they do not rely on an existing standard, Pahl *et al.* [14] use a middleware providing similar feature by communicating predefined classes attached with each device's requests.

Fan *et al.* [21] differ in their strategy with their implementation of FTL. While a global model is trained in the cloud, each client trains a personalized version of this model thanks to the transfer learning (TL) approach. This allows to train models accurately on the singularities of each network, while improving the overall performance of the system. Zhao *et al.* [16] also have a specific approach with MTL, as it allows their model to target different problems at once. In their experiment, the same base model is then trained for anomaly detection, virtual private network (VPN) traffic classification, and TOR

traffic recognition.

In their work, the authors of [23], [32] also train multiple models, but not per class or environment. Segmented Federated Learning (SFL) is a dynamic approach that creates new models depending on the accuracy of the clients. When a client's accuracy is too far from the others, a new branch is created, and some clients' models are aggregated in a new *segmented* cluster.

Qin and Kondo [31] propose another way of building different more specific models, by training models depending on the feature set used by the local device. They emit the hypothesis of building models per attack: devices could train a model for *DoS* attacks, others for *Probes*.

The other works considered in this survey use a global model for their detection [13], [15], [18]–[20], [22], [24]–[28], [30], regardless of the data type, or detection method.

11) Analyzed dataset: While the literature on intrusion detection provides multiple datasets (Table II), the choice of the dataset depends on the use case. In the context of federated intrusion detection, two types of datasets can be found: network traces or sensor values. Table IV shows the comparison between the selected approaches, with the datasets used for training and evaluation. As explained in Section V-C1, two (2) out of twenty-two (22) works are targeted to sensor values [15], [20]. In their paper, Zhang, Lu, *et al.* [20] generate their own dataset using Raspberry Pis as sensors to represent air conditioners. The dataset is labeled and contains seventy features, but only eighteen are kept by the authors, all regarding the status of the devices (*e.g.* evaporator water temperature, condenser water temperature, compressor air temperature, exhaust air temperature, etc.), but the dataset has not been made public, to the best of our knowledge. Schneble *et al.* [15] are using the public dataset MIMIC [87], which contains a collection of medical information, such as arterial blood pressure, and the raw ECG signals corresponding to the measured voltage across leads on the body. The authors extract seven features from the dataset.

The other works, which are using network traffic as source, differ on the device types they consider. Li, Wu, *et al.* [18] use a public dataset [90] made of MODBUS data labeled in eight different classes, the first one for *normal* operation, the other for seven types of cyberattacks. The dataset contains twenty-six features and one label. Qin, Poularakis, *et al.* [26] focus on botnets in one of their scenarios. To test validate their assumptions, they use the ISCX Botnet 2014 [92] dataset, which is composed of diverse botnet and malware traffic. The dataset is replayed to the gateway to capture packet-level features from the gateway directly.

The authors of [19], [27] and [29] use the dataset generated during the KDD Cup 99, which is also public [57]. The dataset contains 41 features and one label, classifying the attacks into either normal traffic, or one of the four represented classes of attacks:

- DoS attacks—high traffic volume from a (or multiple) host to another;
- User to Root (U2R)—privilege escalation to gain access to a root account in the system (password sniffing or guessing, brute force, ...);
- Remote to Local (R2L)—malicious incoming traffic over the network to gain local user access (exploit, ...)
- Probing attacks—information gathering by sending requests (nmap, ...).

The dataset has however been improved in 2009 as NSL-KDD, to cope with the shortcomings of the original version, namely redundant records, and low difficulty [58]. Multiple papers use this NSL-KDD version [9], [21], [22], [24], [25], [31], separated in two datasets Train+ and Test+, which contain 125,973 and 22,544 records, respectively. More recent datasets have also been used for IT use cases, like CICIDS2017 [63] in [16], [21], [26]. Chen, Lv, *et al.* [13] use both, as a way to increase the heterogeneity of input data. They also add the WSN-DS dataset [93] to target wireless communication.

Some works address specifically the issue of IoT devices, which introduce a new whole set of constraints. They have little performance, poor patching capabilities, often weak encryption and authentication mechanisms [1]. Furthermore, their heterogeneity and sporadic traffic make IDS be less efficient and/or inadequate [12], [44]. These characteristics make common dataset inefficient to train detection models tailored for the IoT. As a response to this, both Pahl *et al.* [14] and Nguyen, Marchal, *et al.* [12] generated their own dataset. In [12], the authors generated from 33 "from the shelf" IoT devices (cameras, light bulbs, routers, home assistants...) recorded during their different phases of operation (On, Off, user activity...). Three datasets have been generated:

- *activity*: execute the user interactions 20 times for each device. Some devices are not submitted to user interactions, so the dataset only provides standby traffic.
- *real-life*: 11 devices are deployed in a "smart home" configuration with real users which interacted with the devices over a week
- *attack*: The Mirai malware was deployed in the selected devices to characterize the different phases of the malware (pre-infection, infection, scanning, and DDOS)

The dataset is not published yet[1]. Pahl *et al.* [14] used another approach to generate their dataset, by using the VSL middleware. The dataset has been published on Kaggle [108]. 7 VSL microservices are monitored in four different sites over 24 hours. Two datasets have been created, using different periodicities for their services. The first one contains anomalous accesses, the second DoS attacks. In total, 7 attack classes are represented, which corresponds to 3 percent of the dataset. From the data provided by VSL are extracted 8 features: addresses and types for source and destination (4 features), operation, data type, value, and timestamp. A computed *periodicity* is added afterward to the features (Section V-C1).

While public datasets are not always available, especially recent topics like IoT [12], they ensure the reproducibility of

[1]The authors have been contacted on June 7 (2021), and stated that the anonymization of the data is still in progress.

experiments, and comparison with the state of the art. The data sets generated need to be shared so that the community can validate the results. Reproducibility is currently a major hurdle for FIDS research.

12) Costs and metrics: Researchers use metrics in the literature to assess, validate, and compare their solutions. In this work, metrics are divided in three categories that follow the life cycle of FIDSs: training, federation, and execution.

While training DL models, most ML frameworks use and display training loss and training accuracy, that are used to adapt the model's weights at each epoch. When plotted on a time-based frame (time, epochs, or rounds in case of FL), these metrics show the evolution of the model's training. It can be used to measure the convergence time of the model, often characterized as obtaining an accuracy above a defined threshold (*e.g.* 90% in [13]), or with the percentage of loss improvement between two epochs (*e.g.* 0.01 in [25]). Training time also serve as a comparison between approaches [15], even though it depends a lot on the underlying hardware architecture. Finally, it can be used as a metric to select other hyper-parameters, such as the number of epochs in [29].

Li, Zhou, *et al.* [28] summed training time and communication time at each round in a unique metric *time cost*. This overall training cost is justified by their constrained environment, where resources are very limited. Therefore, instead of reaching for maximum accuracy, the authors fix the accuracy as a target and iterate on hyper- and meta-parameters (*e.g.* learning rate, epochs per round) to find the lowest *time cost*.

Algorithm complexity and resource consumption are also relevant metrics to measure local training costs. Constrained use cases like IoT require complex algorithm to run on resource-limited devices. In [14], the authors also study complexity to choose BIRCH clusters instead of K-means, as updating the former is easier—$\mathcal{O}(d)$ vs $\mathcal{O}(n * d)$, where d is the dataset size.

Hardware-related resources are used by [9], [16], mostly to emphasize differences between their approach and another, often more standard one. These resources often include CPU, disk and memory usage, as well as energy consumption. However, evaluating hardware-related metrics requires experiments to be implemented using the same hardware and software stacks. Hardware- and energy-based metrics are especially relevant in constrained scenarios [12], [15], whereas training time is relevant for most use case, while not a priority. When these measures are collected on reference hardware, it can also be used to evaluate the feasibility of the approach, as in [12], if the hardware matches the deployment constraints of the study.

Federation-related metrics are heavily tied to the communication between clients, or with a server. The communication overhead is a core metric of FIDSs, as high bandwidth consumption is a drawback of CIDSs (see Challenge 4), especially in constrained environments [26]. The overhead is often measured in bytes, either per packets [14], or for the total of all communications [15], [20].

Metrics must be adapted to the specificities of each solution, for instance when adding a feature. Consequently, Zhang, Lu, *et al.* [20] add specific metrics in their evaluation to measure the impact of using the blockchain, like the time of the *blockchain encoring process*. Some works [9], [18], [19], [21]–[24], [30], on the other hand, do not cover federation-related metrics in their evaluation, which is questionable as it is a critical part of FIDSs.

Finally, execution-related metrics are mostly focused on performance, and often come from the ML community. As shown in Table V, *accuracy* is used by almost all reviewed works, followed by *precision* and *recall*. Researchers often use accuracy to compare their results with related works. Accuracy can be completed by *fallout*, *specificity*, and *miss rate* (Section IV-D).

More performance metrics can be derived from these, like *F1-score*, or the area under the curve (AUC). The latter is obtained from the receiver operating characteristic (ROC) curve, which is used to evaluate binary classification algorithms [9], [12], [14]. MCC is another popular metric for binary classification tasks, and considered by some as the best metric for this use case [71]. It is used in [9]. Finally, the confusion matrix is used by [13], [16], [24], [32]. It allows a visualization of classification performance by opposing predicted classes against the real ones. While critical in intrusion detection tasks, the miss rate is never directly addressed by the selected works, as the author often prefer the related *recall* metric.

Other execution metrics like execution time are considered, as it can be critical for intrusion detection tasks. Latency allows a comparison between different architectures, especially *centralized*, *distributed*, and *decentralized* [9]. Latency is also relevant for highly constrained setups, as in [26]. As pointed out in Section V-C3, *ML location* can have an impact on data collection, but also on detection latency, if data need to travel over network to be analyzed.

Execution metrics are only relevant when comparing works that share implementation. Such comparison is often performed by reimplementing a selection of related works. They can also be used to highlight differences between approaches, like between *local*, *federated*, and *ideal* models [9], [18].

VI. Discussion and open issues

This section provides a synthetic overview of the state of the art of FIDS to succinctly answer RQs about components and metrics (RQs 2.1 and 2.2), while a more thorough analysis is proposed in Section V-C. It answers RQ 3.3 by discussing the limitations of this work, and identifies open issues and research directions.

A. State of the Art of FIDS

Even though the set represented by the selected works is small, there are a few conclusions that can be extracted. As denoted in Section IV, FL has been introduced for two reasons: (1) it breaks isolated architectures by allowing learning over distributed data; (2) it speeds up training, and reduces communications when compared to centralized learning. FIDSs,

enable collaborating without endangering each participant's security by keeping data local.

As highlighted in Table IV, most reviewed works use offline supervised classifiers. This makes them less applicable to real-world usage, due to the lack of adaptability and the need for labeled data. However, supervised classifiers are easier to design and deploy, and allow researchers to focus on the federation aspects of FIDSs.

Most works use NNs for detection, regardless of the data on which they are trained. This is in line with the community of ML–based detection that already tends toward NN for their high performance and generalization capabilities [44]. Moreover, the selected set of papers implies that CNN and RNN perform better at generalizing shared models than simpler NNs like MLP. Furthermore, since NNs require large quantities of data and high computing power, their training is time-consuming. Therefore, distributing the training among clients allows producing accurate models in a shorter period of time.

Most architectures in the paper set rely on a gateway to perform the analysis. It is in fact simpler to deploy and convenient as it reduces the number of probes. This also limits these approaches to NIDS use, as only network characteristics are visible at the gateway. It omits other aspects of the systems, such as local device-related metrics, like processes, inter-process communication, or memory consumption, that could be useful for correlation purposes. Moreover, the closer the learning is made to devices, the more numerous, and therefore less generalized, the models will be. In fact, all *meta-parameters* an impact the performance of the final system. *Meta-parameters* are configuration variable that are not related to the model itself, such as the architecture, the aggregation strategy, or the use of device classes. Especially, *model target* (Section V-C10) and *aggregation strategy* (Section V-C9) can significantly affect the performance.

Performance can be characterized solely with ML indicators. Table V shows the ML approaches used by the papers and their performance. Common metrics include *accuracy*, *precision*, *recall*, *fallout*, and *F1-score*. These metrics alone do not allow comparing works between them, because of their too important differences, *e.g.* dataset used, architecture, use of personalized models. They are however useful to compare one solution with an ideal system with unconstrained resources and access two all data at once, to see how the federation affect the model's performance. Depending on the application of FIDS, other metrics can be used, like the resource consumption on constrained devices, the latency of the detection system when performed on a dedicated remote device, or the communication overhead.

While we observe a convergence in metrics for performance measurements, these metrics comes from research on ML-based detection systems [36], [42], [44]. Other metrics depend on objectives and application, and are not consistent among FIDSs evaluations.

Finally, publication is so far use-case–based, and some contributions in the selected works rely primarily on applying FIDS to a use case for the first time. As interest for FIDSs grows, we expect contributions to tend towards the development of methods and techniques, instead of use cases.

B. Limitations of this survey

This survey reviewed 22 technical papers about FIDSs, selected using SLR methodology. This ensures that the selected papers are representative of existing works in this field. Other surveys in similar but broader fields worked with bigger quantities of papers; 231 in [50] about FL, or 95 in [45] for ML-based IDSs. Therefore, all conclusion extracted from the selected works must be put in perspective of the number of analyzed papers.

Furthermore, SLR methodology guarantees the exhaustive aspect of the selection. However, relevant papers may have been missed; especially, edge–use-cases and unusual wording can exclude papers from the selection process. We expect the steps presented in Section III to mitigate this risk, notably snowballing.

Moreover, the selected metrics give insight on the quality of the predictions, and more importantly the comparison between FIDS and local detection, when provided. As the selected works target different use cases with different objectives, a performance metric-based comparison is less relevant. Using the same datasets, hardware and network configuration, and coding frameworks, a thorough reimplementation of the reviewed papers could provide significant contributions.

C. Open issues and research directions

As FL is becoming more mature, new research tend focus either on side-aspects like security and privacy [8], [38], [67], [109], [110], or on its application to a specific use case, as do the works selected in this survey. This survey focuses on the application of FL to security and intrusion detection. Therefore, after the analyses in Section V, the selection emphasizes on algorithms that are: (i) efficient to classify and group device's behavior; (ii) have models that can be easily aggregated; (iii) select relevant features that keep their significance upon aggregation; (iv) can cope with the heterogeneity of monitored environments and devices.

This section reviews open questions identified by literature, and according research directions.

1) System performance: Like any detection system, FIDS look for *absolute* performance: a system with perfect classification score, yielding zero false positives or negatives. To that end, several leads have been identified by the literature. Firstly, Generative Adversarial Networks (GAN) are to be studied as inputs to the federated system [15]. GAN are used to train an ML classifier by generating false examples with another dedicated network, with the objective of allowing the classifier to detect adversarial examples.

Furthermore, as pointed out in Section V-C2, the impact of feature selection on a model's performance is undisputed. However, future work is required to properly identify features that are especially suited for FL and knowledge sharing. Extending relevant features is also to be addressed, thus closing the gap between NIDS and HIDS use.

Another consequent part of training efficient ML models depends on build good behavior representation. Tools have been proposed in the literature like periodicity mining [12], [14], but they face limitations. Periodicity mining do perform well on simple environments like autonomous IoT devices, but falters on more complex behaviors including human interaction or chaotic traffic. Furthermore, protocol mining could provide good performance in IoT and IIoT applications by allowing to characterize correct operation, *e.g.* preventing a motor from driving longer than its specification allows [14]. More generally, manual feature selection is an important part of the identified potential improvements [32], especially for applicability in other contexts.

Finally, other works propose to work on measuring the influence of model's hyperparameters on the performance of the model [23], *e.g.* number of epochs, learning rate, batch size. This is in accordance with the findings of this survey. The link between the system's performance and its hyper- and meta-parameters is not explicit. Future works are still required to evaluate and compare each meta-parameter to understand their impact on performance.

2) Adaptability and scalability: Distributed systems such as FL are often used to cope with resource limitation, notably in terms of computation and bandwidth. However, and as pointed out by several selected works [9], [21], FL faces limitations when dealing with a high number of clients. Visible in the literature, participant selection for each round is a known strategy.

Nevertheless, current approaches have a lot of limitations, including devices dropout during the process, long response time for the upload and update of the models, clients with less relevant data [22]. Further research is therefore required on client selection [21] in the context of FIDS: score- or time-based dynamic fusion [78], reputation, number of attacks detected, and so on.

Moreover, constrained environments like low-bandwidth networks, or low-powered devices, may also impact the ability of FL to provide detection in a timely fashion (Section IV-B). Further work is thus required on improving the performance of FL by implementing compression algorithm [21] or globally reducing the number of computation rounds [22]. These constraints also accentuate the heterogeneity of environment and devices.

3) Transferability: Current solutions tend to focus on federating learning and detection for same-domain devices and resources. Hence, open issues include allowing the federation of cross-domain clients [18]. As pointed out in Section V-C1, the features selected for model training have to be applicable to multiple environments. Transfer learning [49], [111] and its federated variant FTL [21], [112] have been applied to similar domains in the past, and might also represent a favorable direction for future research in terms of adaptability. Since the submission of this study, three papers [113]–[115] have been published in this direction, highlighting the relevance of this topic for the community.

While generating trained models per class of devices has been experimented-on in literature [12], [14], current methods often consider that all local models need to share the same NN architecture and hyperparameters. This limitation makes current FIDS less versatile and transportable to other domains. Multiple techniques could be considered to overcome these issues. Allowing the training of multiple variations of the same models could provide better adaptability [19]. Following this idea, systems could provide multiple algorithms with models tailored for specific use cases, allowing better local results. Works on the balance between providing more models or training them more could provide significant contributions. Finally, FTL could also be applicable to generate use-case–specific models from the experience of the global federated system, with only few local data.

4) Security and timeliness: Using FL or ML to detect intrusions can introduce new threats to the system, like model poisoning. Several works have reviewed vulnerabilities of FL systems and proposed counter-measures [8], [38], [109], [116]. With FL, model poisoning becomes easier, as one participant can theoretically impact the model of every other. FLGUARD [76] implements model-poisoning detection, but other strategies could be studied, especially in the aspects of reputation systems, and weighting of the aggregation. Likewise, current solutions require an increase of the trustworthiness of client devices for the aggregation process [20], inspired by the state of the art of collaboration systems and information-sharing platforms. In particular, current research on these topics address problems such as trust or reputations [2], [39], which are relevant for FIDS.

Moreover, as attacks evolve, the training data tend to be easily outdated. Updating strategies need to be studied to provide accurate results as time goes [21], and adapt to changes in the traffic behavior [31]. While the security of the parameter aggregation has been tackled with homomorphic encryption [18], [28], Chen, Lv, *et al.* [13] identify this aspect as potential future works, as well as other security measures as MPC or differential privacy.

5) Self-defense and self-healing: As highlighted in Section V-C5, current research on FIDS is focused on intrusion detection and attack classification. Mitigation is barely represented in the literature [9]. However, technologies like SDN offer quick mitigation capabilities, and recent works study the effectiveness of such defense mechanisms [117], [118]. New emerging applications like self-defense and self-healing systems could benefit from FIDS and other FL-based technologies.

VII. Conclusion

This paper provided the first Systematic Literature Review (SLR) on FL-based intrusion detection and mitigation. While FL is maturing, using it for intrusion detection and mitigation adds new challenges. Section V resented successful applications of FIDSs. By comparing relevant state of the art, this work laid the ground for assessing and comparing future works on machine-learning based Federated Intrusion

Detection Systems (FIDSs): Section V-A provided a reference architecture and a taxonomy.

Section V-B identified the major venues, the most active research groups, and the evolution of the FL for intrusion detection and mitigation. It reveals an increasing interest of the community in the topic. Section V-C reviewed central papers by applying the taxonomy, see Figure 4. The focus was on NNs, their aggregation and detection capabilities, and the impact that hyper- and meta-parameters have on their performance that is critical for FIDS. Finally, Section VI-C revealed topics that need future research. It highlights limitations of current works on model performance, behavior modeling, adaptability, and security.

Overall, this work showed how FL can improve IDSs, making it an important technique for increasing the resilience of future systems. While enabling information-sharing, FIDSs bring the strong advantage of preserving privacy and security by not sharing raw data but only models that abstract from privacy-critical raw-data by design.

Acronyms

FL Federated Learning.
ML Machine Learning.
IDS Intrusion Detection System.
IT Information Technology.
OT Operational Technology.
TI Threat Intelligence.
CIDS Collaborative IDS.
FIDS Federated Intrusion Detection System.
RQ research question.
CTI Cyber Threat Intelligence.
AI Artificial Intelligence.
DDoS Distributed Denial of Service.
IoT Internet of Things.
IID Independent and Identically Distributed.
SLR Systematic Literature Review.
APT Advanced Persistent Threat.
SPoF single point-of-failure.
CD-FL Cross-device Federated Learning.
CS-FL Cross-silo Federated Learning.
HFL horizontal federated learning.
VFL vertical federated learning.
FTL federated transfer learning.
PPV positive predictive value.
TPR true positives rate.
TNR true negative rate.
FPR false positives rate.
FNR false negative rate.
MCC Mathew Correlation Coefficient.
MAPE-K Monitor-Analyze-Plan-Execute plus Knowledge.
IIoT Industrial Internet of Things.
MPC multi-party computation.
HIDS Host-based Intrusion Detection System.
NIDS Network-based Intrusion Detection System.
VSL Virtual State Layer.
CPS Cyber-Physical System.
AV Autonomous Vehicles.
STIN Satellite-Terrestrial Integrated Network.
IAT inter-arrival time.
PCA Principal Component Analysis.
MEC Mobile Edge-Computing.
SDN Software-defined networking.
BNN Binarized Neural Network.
DoS Denial of Service.
P2P peer-to-peer.
NN Neural Network.
RNN Recurrent Neural Network.
GRU Gated Recurrent Unit.
SVM Support Vector Machine.
CNN Convolutional Neural Network.
MLP Multilayer Perceptron.
DL Deep Learning.
GMM Gaussian Mixture Model.
ELM Extreme Learning Machine.
MTL Multi-Task Learning.
DT Decision Tree.
RF Random Forest.
KNN K-nearest neighbors.
SGD Stochastic Gradient Descent.
RSU roadside unit.
V2X Vehicle-to-Everything.
FML Federated Mimic Learning.
CDW Centroid Distance Weighted.
DQN deep Q-network.
RL Reinforcement Learning.
BIRCH balanced iterative reducing and clustering using hierarchies.
μS microservice.
MUD Manufacturer Usage Description.
TL transfer learning.
VPN virtual private network.
SFL Segmented Federated Learning.
AUC area under the curve.
ROC receiver operating characteristic.
GAN Generative Adversarial Networks.

References

[1] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations," *IEEE Communications Surveys & Tutorials*, 2019.

[2] T. D. Wagner, K. Mahbub, E. Palomar, and A. E. Abdallah, "Cyber threat intelligence sharing: Survey and research directions," *Computers & Security*, 2019.

[3] ENISA, "Incentives and Challenges for Information Sharing in the Context of Network and Information Security," 2010.

[4] ——, "Exploring the opportunities and limitations of current Threat Intelligence Platforms," 2017.

[5] *Directive (EU) 2016/1148 of 6 July 2016 concerning measures for a high common level of security of network and information systems across the Union*, in collab. with The European Parliament and The Counsil, 2016.

[6] ——, "Actionable Information for Security Incident Response," 2014.

[7] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," 17, 2016.

[8] T. D. Nguyen, P. Rieger, M. Miettinen, and A.-r. Sadeghi, "Poisoning Attacks on Federated Learning-based IoT Intrusion Detection System," in *The Network and Distributed System Security Symposium (NDSS) 2020*, 2020.

[9] S. Rathore, B. Wook Kwon, and J. H. Park, "BlockSecIoTNet: Blockchain-based decentralized security architecture for IoT network," *Journal of Network and Computer Applications*, December 2018 2019.

[10] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated Machine Learning: Concept and Applications," *ACM Transactions on Intelligent Systems and Technology*, 28, 2019.

[11] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konečný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and Open Problems in Federated Learning," 8, 2021, (visited on 04/01/2022).

[12] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A Federated Self-learning Anomaly Detection System for IoT," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, IEEE, 2019.

[13] Z. Chen, N. Lv, P. Liu, Y. Fang, K. Chen, and W. Pan, "Intrusion Detection for Wireless Edge Networks Based on Federated Learning," *IEEE Access*, 2020. (visited on 10/25/2021).

[14] M.-O. Pahl and F. X. Aubet, "All Eyes on You: Distributed Multi-Dimensional IoT Microservice Anomaly Detection," *14th International Conference on Network and Service Management, CNSM 2018 and Workshops, 1st International Workshop on High-Precision Networks Operations and Control, HiPNet 2018 and 1st Workshop on Segment Routing and Service Function Chaining, SR+SFC 2*, Cnsm 2018.

[15] W. Schneble and G. Thamilarasu, "Attack detection using federated learning in medical cyber-physical systems," presented at the International Conference on Computer Communications and Networks, 2019.

[16] Y. Zhao, J. Chen, D. Wu, J. Teng, and S. Yu, "Multi-Task Network Anomaly Detection using Federated Learning," in *Proceedings of the Tenth International Symposium on Information and Communication Technology - SoICT 2019*, Hanoi, Ha Long Bay, Viet Nam: ACM Press, 2019. (visited on 06/07/2021).

[17] B. Cetin, A. Lazar, J. Kim, A. Sim, and K. Wu, "Federated Wireless Network Intrusion Detection," in *2019 IEEE International Conference on Big Data (Big Data)*, Los Angeles, CA, USA: IEEE, 2019. (visited on 10/25/2021).

[18] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "DeepFed: Federated Deep Learning for Intrusion Detection in Industrial Cyber-Physical Systems," *IEEE Transactions on Industrial Informatics*, 2020.

[19] Y. Chen, J. Zhang, and C. K. Yeo, "Network Anomaly Detection Using Federated Deep Autoencoding Gaussian Mixture Model," in *Machine Learning for Networking*, ser. Lecture Notes in Computer Science, S. Boumerdassi, É. Renault, and P. Mühlethaler, Eds., Cham: Springer International Publishing, 2020. (visited on 06/07/2021).

[20] W. Zhang, Q. Lu, Q. Yu, Z. Li, Y. Liu, S. K. Lo, S. Chen, X. Xu, and L. Zhu, "Blockchain-based Federated Learning for Device Failure Detection in Industrial IoT," *IEEE Internet of Things Journal*, 6, 2020.

[21] Y. Fan, Y. Li, M. Zhan, H. Cui, and Y. Zhang, "IoTDefender: A Federated Transfer Learning Intrusion Detection Framework for 5G IoT," in *2020 IEEE 14th International Conference on Big Data Science and Engineering (BigDataSE)*, Guangzhou, China: IEEE, 2020. (visited on 10/04/2021).

[22] S. A. Rahman, H. Tout, C. Talhi, and A. Mourad, "Internet of Things Intrusion Detection: Centralized, On-Device, or Federated Learning?" *IEEE Network*, 2020. (visited on 06/01/2021).

[23] Y. Sun, H. Ochiai, and H. Esaki, "Intrusion Detection with Segmented Federated Learning for Large-Scale Multiple LANs," in *2020 International Joint Conference on Neural Networks (IJCNN)*, Glasgow, United Kingdom: IEEE, 2020. (visited on 10/01/2021).

[24] N. A. A.-A. Al-Marri, B. S. Ciftler, and M. Abdallah, "Federated Mimic Learning for Privacy Preserving Intrusion Detection," 13, 2020, (visited on 10/25/2021).

[25] S. Kim, H. Cai, C. Hua, P. Gu, W. Xu, and J. Park, "Collaborative Anomaly Detection for Internet of Things based on Federated Learning," in *2020 IEEE/CIC International Conference on Communications in China (ICCC)*, Chongqing, China: IEEE, 9, 2020. (visited on 10/25/2021).

[26] Q. Qin, K. Poularakis, K. K. Leung, and L. Tassiulas, "Line-Speed and Scalable Intrusion Detection at the Network Edge via Federated Learning," 2020.

[27] X. Hei, X. Yin, Y. Wang, J. Ren, and L. Zhu, "A trusted feature aggregator federated learning for distributed malicious attack detection," *Computers & Security*, 2020. (visited on 10/25/2021).

[28] K. Li, H. Zhou, Z. Tu, W. Wang, and H. Zhang, "Distributed Network Intrusion Detection System in Satellite-Terrestrial Integrated Networks Using Federated Learning," *IEEE Access*, 2020. (visited on 10/25/2021).

[29] H. Liu, S. Zhang, P. Zhang, X. Zhou, X. Shao, G. Pu, and Y. Zhang, "Blockchain and Federated Learning for Collaborative Intrusion Detection in Vehicular Edge Computing," *IEEE Transactions on Vehicular Technology*, 2021. (visited on 10/04/2021).

[30] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT Edge Devices," *IEEE Internet of Things Journal*, 2021. (visited on 10/01/2021).

[31] Y. Qin and M. Kondo, "Federated Learning-Based Network Intrusion Detection with a Feature Selection Approach," in *2021 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, Kuala Lumpur, Malaysia: IEEE, 12, 2021. (visited on 10/04/2021).

[32] Y. Sun, H. Esaki, and H. Ochiai, "Adaptive Intrusion Detection in the Networking of Large-Scale LANs With Segmented Federated Learning," *IEEE Open Journal of the Communications Society*, 2021. (visited on 10/04/2021).

[33] S. Agrawal, S. Sarkar, O. Aouedi, G. Yenduri, K. Piamrat, S. Bhattacharya, P. K. R. Maddikunta, and T. R. Gadekallu, "Federated Learning for Intrusion Detection System: Concepts, Challenges and Future Directions," 16, 2021, (visited on 12/02/2021).

[34] E. M. Campos, P. F. Saura, A. González-Vidal, J. L. Hernández-Ramos, J. B. Bernabe, G. Baldini, and A. Skarmeta, "Evaluating Federated Learning for Intrusion Detection in Internet of Things: Review and Challenges," 2, 2021, (visited on 12/02/2021).

[35] M. Alazab, S. P. R M, P. M, P. Reddy, T. R. Gadekallu, and Q.-V. Pham, "Federated Learning for Cybersecurity: Concepts, Challenges and Future Directions," *IEEE Transactions on Industrial Informatics*, 2021. (visited on 12/02/2021).

[36] O. Faraj, D. Megías, A.-M. Ahmad, and J. Garcia-Alfaro, "Taxonomy and challenges in machine learning-based approaches to detect attacks in the internet of things," in *Proceedings of the 15th International Conference on Availability, Reliability and Security*, Virtual Event Ireland: ACM, 25, 2020. (visited on 06/23/2021).

[37] E. Vasilomanolakis, S. Karuppayah, and M. Fischer, "Taxonomy and Survey of Collaborative Intrusion Detection," *ACM Computing Surveys*, 2015.

[38] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and

privacy of federated learning," *Future Generation Computer Systems*, 2021.

[39] F. Skopik, G. Settanni, and R. Fiedler, "A problem shared is a problem halved: A survey on the dimensions of collective cyber defense through security information sharing," *Computers & Security*, 2016.

[40] W. Tounsi and H. Rais, "A survey on technical threat intelligence in the age of sophisticated cyber attacks," *Computers & Security*, 2018.

[41] A. Pala and J. Zhuang, "Information Sharing in Cybersecurity: A Review," *Decision Analysis*, 2019.

[42] A. L. Buczak and E. Guven, "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," *IEEE Communications Surveys Tutorials*, 2016.

[43] W. Meng, E. W. Tischhauser, Q. Wang, Y. Wang, and J. Han, "When Intrusion Detection Meets Blockchain Technology: A Review," *IEEE Access*, 2018.

[44] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network Intrusion Detection for IoT Security Based on Learning Techniques," *IEEE Communications Surveys & Tutorials*, 2019.

[45] K. A. da Costa, J. P. Papa, C. O. Lisboa, R. Munoz, and V. H. C. de Albuquerque, "Internet of Things: A survey on machine learning-based intrusion detection approaches," *Computer Networks*, 2019.

[46] C. V. Zhou, C. Leckie, and S. Karunasekera, "A survey of coordinated attacks and collaborative intrusion detection," *Computers & Security*, 2010. (visited on 07/21/2021).

[47] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Access*, 2020.

[48] L. Lyu, H. Yu, and Q. Yang, "Threats to Federated Learning: A Survey," *arXiv*, 4, 2020.

[49] S. Shen, T. Zhu, D. Wu, W. Wang, and W. Zhou, "From Distributed Machine Learning To Federated Learning: In The View Of Data Privacy And Security," *Concurrency and Computation: Practice and Experience*, 23, 2020. arXiv: 2010.09258. (visited on 10/04/2021).

[50] S. K. Lo, Q. Lu, C. Wang, H.-Y. Paik, and L. Zhu, "A Systematic Literature Review on Federated Machine Learning: From A Software Engineering Perspective," *ACM Computing Surveys*, 2021. arXiv: 2007.11354. (visited on 10/04/2021).

[51] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," 2007.

[52] R. E. Navas, F. Cuppens, N. B. Cuppens, L. Toutain, and G. Z. Papadopoulos, "MTD, Where Art Thou? A Systematic Review of Moving Target Defense Techniques for IoT," *IEEE Internet of Things Journal*, 2020.

[53] S. Keshav, "How to read a paper," *acm special interest group on data communication*, 2007.

[54] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, 2019. (visited on 03/11/2022).

[55] H. Liu and B. Lang, "Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey," *Applied Sciences*, 17, 2019. (visited on 03/11/2022).

[56] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security*, 2009.

[57] SigKDD. (1999). "KDD Cup 1999 Dataset," (visited on 04/26/2021).

[58] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, IEEE, 2009.

[59] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion Detection in 802.11 Networks: Empirical Evaluation of Threats and a Public Dataset," *IEEE Communications Surveys & Tutorials*, 2016.

[60] M. Ring, S. Wunderlich, D. Grüdl, D. Landes, and A. Hotho, "Creation of Flow-Based Data Sets for Intrusion Detection," *Journal of Information Warfare*, 2017. JSTOR: 26504117.

[61] ——, "Flow-based benchmark data sets for intrusion detection," *Proceedings of the 16th European Conference on Cyber Warfare and Security (ECCWS)*, 2017.

[62] S. A. V. Jatti and V. J. Kishor Sontif, "UNSW-NB15: A Comprehensive Data set for Network Intrusion Detection Systems," *International Journal of Recent Technology and Engineering*, 2S11 2, 2019.

[63] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization:" in *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, Funchal, Madeira, Portugal: SCITEPRESS - Science and Technology Publications, 2018. (visited on 10/14/2021).

[64] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics," *IEEE Transactions on Mobile Computing*, 1, 2019. (visited on 05/21/2021).

[65] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated Learning: Strategies for Improving Communication Efficiency," 18, 2016.

[66] J. Ren, H. Wang, T. Hou, S. Zheng, and C. Tang, "Federated Learning-Based Computation Offloading Optimization in Edge Computing-Supported Internet of Things," *IEEE Access*, 2019.

[67] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical Secure Aggregation for Privacy-Preserving Machine Learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, New York, NY, USA: ACM, 30, 2017.

[68] U. Majeed and C. S. Hong, "FLchain: Federated Learning via MEC-enabled Blockchain Network," in *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, IEEE, 2019.

[69] C. M. Bishop, *Pattern Recognition and Machine Learning*, ser. Information Science and Statistics. New York: Springer, 2006, 738 pp.

[70] D. Hand and P. Christen, "A note on using the F-measure for evaluating record linkage algorithms," *Statistics and Computing*, 2018. (visited on 11/10/2021).

[71] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, 2020. (visited on 03/11/2022).

[72] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, 2003.

[73] A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, B.-J. (Hsu, and K. Wang, "An Overview of Microsoft Academic Service (MAS) and Applications," in *Proceedings of the 24th International Conference on World Wide Web*, Florence Italy: ACM, 18, 2015. (visited on 10/22/2021).

[74] R. Doshi, N. Apthorpe, and N. Feamster, "Machine Learning DDoS Detection for Consumer Internet of Things Devices," *2018 IEEE Security and Privacy Workshops (SPW)*, Ml 11, 2018.

[75] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne, "Private federated learning on

vertically partitioned data via entity resolution and additively homomorphic encryption," 28, 2017, (visited on 10/04/2021).

[76] T. D. Nguyen, P. Rieger, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, A.-R. Sadeghi, T. Schneider, and S. Zeitouni, "FLGUARD: Secure and Private Federated Learning," 21, 2021, (visited on 05/18/2021).

[77] M. Kim, O. Gunlu, and R. F. Schaefer, "Federated Learning with Local Differential Privacy: Trade-offs between Privacy, Utility, and Communication," 2021.

[78] W. Zhang, T. Zhou, Q. Lu, X. Wang, C. Zhu, H. Sun, Z. Wang, S. K. Lo, and F.-Y. Wang, "Dynamic Fusion based Federated Learning for COVID-19 Detection," *arXiv*, 22, 2020.

[79] (). "Conference portal - CORE."

[80] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards Federated Learning at Scale: System Design," *arXiv*, 4, 2019.

[81] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated Optimization: Distributed Machine Learning for On-Device Intelligence," 8, 2016.

[82] S. Marchal, M. Miettinen, T. D. Nguyen, A.-R. Sadeghi, and N. Asokan, "AuDI: Toward Autonomous IoT Device-Type Identification Using Periodic Communication," *IEEE Journal on Selected Areas in Communications*, 2019. (visited on 06/04/2021).

[83] C. Kruegel and T. Toth, "Using Decision Trees to Improve Signature-Based Intrusion Detection," in *Recent Advances in Intrusion Detection*, 2003.

[84] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, 2009. (visited on 03/20/2022).

[85] M.-y. Zhu, Z. Chen, K.-f. Chen, N. Lv, and Y. Zhong, "Attention-based federated incremental learning for traffic classification in the Internet of Things," *Computer Communications*, 2022. (visited on 01/31/2022).

[86] O. Aouedi, K. Piamrat, G. Muller, and K. Singh, "Intrusion detection for Softwarized Networks with Semi-supervised Federated Learning," 2022.

[87] A. E. Johnson, T. J. Pollard, L. Shen, L.-w. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, and R. G. Mark, "MIMIC-III, a freely accessible critical care database," *Scientific Data*, 24, 2016.

[88] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of Encrypted and VPN Traffic using Time-related Features:" in *Proceedings of the 2nd International Conference on Information Systems Security and Privacy*, Rome, Italy: SCITEPRESS - Science, 2016. (visited on 10/15/2021).

[89] A. Habibi Lashkari, G. Draper Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of Tor Traffic using Time based Features:" in *Proceedings of the 3rd International Conference on Information Systems Security and Privacy*, Porto, Portugal: SCITEPRESS - Science and Technology Publications, 2017. (visited on 10/15/2021).

[90] T. Morris and W. Gao, "Industrial Control System Traffic Data Sets for Intrusion Detection Research," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, ser. Lecture Notes in Computer Science, E. Bayro-Corrochano and E. Hancock, Eds., Cham: Springer International Publishing, 2014. (visited on 06/10/2021).

[91] O. Hideya, "LAN-Security Monitoring Project," 2018.

[92] E. Biglar Beigi, H. Hadian Jazi, N. Stakhanova, and A. A. Ghorbani, "Towards effective feature selection in machine learning-based botnet detection approaches," in *2014 IEEE Conference on Communications and Network Security*, San Francisco, CA, USA: IEEE, 2014. (visited on 10/25/2021).

[93] I. Almomani, B. Al-Kasasbeh, and M. AL-Akhras, "WSN-DS: A Dataset for Intrusion Detection Systems in Wireless Sensor Networks," *Journal of Sensors*, 2016. (visited on 10/25/2021).

[94] J. W. Haines, R. P. Lippmann, D. Fried, M. A. Zissman, E. Tran, and S. B. Boswell, "1999 DARPA Intrusion Detection Evaluation: Design and Procedures," 2001.

[95] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, 2019. (visited on 10/23/2021).

[96] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, D. Breitenbacher, A. Shabtai, and Y. Elovici, "N-BaIoT: Network-based Detection of IoT Botnet Attacks Using Deep Autoencoders," *IEEE Pervasive Computing*, 2018. arXiv: 1805.03409. (visited on 10/23/2021).

[97] M. Tsukada, M. Kondo, and H. Matsutani, "A Neural Network-Based On-device Learning Anomaly Detector for Edge Devices," *IEEE Transactions on Computers*, 2020. (visited on 10/23/2021).

[98] D. Ghosh, R. Sharman, H. Raghav Rao, and S. Upadhyaya, "Self-healing systems — survey and synthesis," *Decision Support Systems*, 2007. (visited on 03/31/2022).

[99] M. Elsadig and A. Abduallah, "Biological Intrusion Prevention and Self-Healing Model for Network Security," in *2010 Second International Conference on Future Networks*, Sanya, Hainan: IEEE, 2010. (visited on 03/31/2022).

[100] J. Ali-Tolppa, S. Kocsis, B. Schultz, L. Bodrog, and M. Kajo, "SELF-HEALING AND RESILIENCE IN FUTURE 5G COGNITIVE AUTONOMOUS NETWORKS," in *2018 ITU Kaleidoscope: Machine Learning for a 5G Future (ITU K)*, Santa Fe: IEEE, 2018. (visited on 03/31/2022).

[101] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick, "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, New York, NY, USA: ACM, 23, 2018.

[102] P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," in *Advances in Cryptology — EUROCRYPT '99*, ser. Lecture Notes in Computer Science, J. Stern, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 1999. (visited on 07/05/2021).

[103] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, ser. Adaptive Computation and Machine Learning. Cambridge, Massachusetts: The MIT Press, 2016, 775 pp.

[104] W. Sun, S. Lei, L. Wang, Z. Liu, and Y. Zhang, "Adaptive Federated Learning and Digital Twin for Industrial Internet of Things," 31, 2020, (visited on 10/04/2021).

[105] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 8, 2014.

[106] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signSGD: Compressed optimisation for non-convex problems," in *Proceedings of the 35th International Conference on Machine Learning*, J. Dy and A. Krause, Eds., ser. Proceedings of Machine Learning Research, PMLR, 10–Jul. 15, 2018.

[107] E. Lear, R. Droms, and D. Romascanu, *Manufacturer usage description specification*, RFC 8520, 2019.

[108] *Ds2ostraffictraces - Kaggle*.

[109] C. Fung, C. J. M. Yoon, and I. Beschastnikh, "Mitigating Sybils in Federated Learning Poisoning," *arXiv*, 14, 2018.
[110] Y. Dong, X. Chen, L. Shen, and D. Wang, "EaSTFLy: Efficient and secure ternary federated learning," *Computers & Security*, 2020. (visited on 05/18/2021).
[111] S. Saha and T. Ahmad, "Federated Transfer Learning: Concept and applications," 6, 2021, (visited on 10/04/2021).
[112] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "FedHealth: A Federated Transfer Learning Framework for Wearable Healthcare," *IEEE Intelligent Systems*, 2020.
[113] Y. Otoum, Y. Wan, and A. Nayak, "Federated Transfer Learning-Based IDS for the Internet of Medical Things (IoMT)," in *2021 IEEE Globecom Workshops (GC Wkshps)*, Madrid, Spain: IEEE, 2021. (visited on 01/31/2022).
[114] T. V. Khoa, D. T. Hoang, N. L. Trung, C. T. Nguyen, T. T. T. Quynh, D. N. Nguyen, N. V. Ha, and E. Dutkiewicz, "Deep Transfer Learning: A Novel Collaborative Learning Model for Cyberattack Detection Systems in IoT Networks," 2, 2021, (visited on 01/31/2022).
[115] Y. Cheng, J. Lu, D. Niyato, B. Lyu, J. Kang, and S. Zhu, "Federated Transfer Learning With Client Selection for Intrusion Detection in Mobile Edge Computing," *IEEE Communications Letters*, 2022. (visited on 01/31/2022).
[116] C. Fung, C. J. M. Yoon, and I. Beschastnikh, "The Limitations of Federated Learning in Sybil Settings," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2020)*, San Sebastian: {USENIX} Association, 2020.
[117] S. S. Bhunia and M. Gurusamy, "Dynamic attack detection and mitigation in IoT using SDN," in *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, Melbourne, VIC: IEEE, 2017. (visited on 04/09/2022).
[118] J. Singh and S. Behal, "Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions," *Computer Science Review*, 2020. (visited on 04/09/2022).

Léo Lavaur is a PhD student in cybersecurity at the engineering school of IMT Atlantique, Rennes, France, and the chair on Cybersecurity in Critical Networked Infrastructures (Cyber CNI). Prior to his admission, he received an engineering degree in information security from the National Engineering School of South Britany (ENSIBS), Vannes, France, in 2020. During his studies, he also worked in industry at Orange Cyberdefense as part-time employee for three years, where he worked on application security, and Wi-Fi rogue access-point detection and location.

Léo's current research focuses on federated architectures, intrusion detection, and machine learning. He studies the collaboration in security systems, and how to share data without compromising security. With the chair and its test bed, he works on building a distributed observatory for attack detection, characterization, and mitigation.

Marc-Oliver Pahl is Professor at IMT Atlantique in Rennes, Brittany, France. He heads the Chair of Cybersecurity in Critical Networked Infrastructures (Cyber CNI) hosted at IMT Atlantique. He is an adjunct professor of Carleton University in Canada. He supervises 3 PhD students at Technical University of Munich, Germany.

Marc-Oliver is Vice President of the German Chapter of the Association for Computing Machinery (ACM), he coordinates the Future Education activities of the German-French Academy for the Industry of the Future.

Marc-Oliver's research focus is on a holistic approach to cybersecurity. He works on security-by-design, anomaly detection, human-in-the-loop, and automation. His goal is making cybersecurity manageable, resulting in highly resilient and reliable systems. Marc-Oliver publishes regularly in the network and service management and security communities.

Yann Busnel is Full Professor at IMT Atlantique. Since January 2017, he is also head of SRCD department (Network Systems, Cybersecurity and Digital Law), on Rennes Campus. He is also in charge of the D2-NTS department (Networks, Telecommunications and Services) of UMR IRISA Research Laboratory.

After his PhD in Computer Sciences at University of Rennes (France) in November 2008, he spent one year at "La Sapienza" University in Roma, Italy (2008-2009). Then, he has been assistant professor (Maître de Conférences) at Université de Nantes, in the Computer Science Department (2009-2014). Following, he has been associate-professor at ENSAI (National Graduate School on Statistics and Data Analysis), where he has acted as Head of the Computer Sciences department (2014-2016). He finally obtained his "Habilitation à diriger les recherches" (HDR) in Computer Sciences from École Normale Supérieure de Rennes (France) in December 2016.

His research topics are mainly related to Data stream analysis over massive data and Models for large-scale distributed systems and networks.

Fabien Autrel Fabien Autrel is a research engineer at IMT Atlantique. He obtained his PhD on the subject of alert fusion, weighted correlation and reaction in a cooperative intrusion detection process in March 2005 at the ONERA research center of Toulouse.

During his postdoctoral fellowship in the Telecom-Bretagne engineering school in 2006 in Rennes, he worked with the Exaprotect company for two years to integrate the research results obtained during his PhD in their EAS product.

Since then, Fabien has continued working on the topic of intrusion detection but also contributed in the field of the formal specification of security policy. More precisely he worked on the OrBAC model and its implementation. More recently, he has been focusing on the cybersecurity of industrial systems.