DeepFed: Federated Deep Learning for Intrusion Detection in Industrial Cyber–Physical Systems

Beibei Li, *Member, IEEE*, Yuhao Wu, Jiarui Song, Rongxing Lu, *Senior Member, IEEE*, Tao Li, *Member, IEEE*, and Liang Zhao, *Member, IEEE*

***Abstract*—The rapid convergence of legacy industrial infrastructures with intelligent networking and computing technologies (e.g., 5G, software-defined networking, and artificial intelligence), have dramatically increased the attack surface of industrial cyber–physical systems (CPSs). However, withstanding cyber threats to such large-scale, complex, and heterogeneous industrial CPSs has been extremely challenging, due to the insufficiency of high-quality attack examples. In this article, we propose a novel federated deep learning scheme, named DeepFed, to detect cyber threats against industrial CPSs. Specifically, we first design a new deep learning-based intrusion detection model for industrial CPSs, by making use of a convolutional neural network and a gated recurrent unit. Second, we develop a federated learning framework, allowing multiple industrial CPSs to collectively build a comprehensive intrusion detection model in a privacy-preserving way. Further, a Paillier cryptosystem-based secure communication protocol is crafted to preserve the security and privacy of model parameters through the training process. Extensive experiments on a real industrial CPS dataset demonstrate the high effectiveness of the proposed DeepFed scheme in detecting various types of cyber threats to industrial CPSs and the superiorities over state-of-the-art schemes.**

***Index Terms*—Data privacy, deep learning, federated learning, industrial cyber–physical system (CPS), intrusion detection.**

Manuscript received June 30, 2020; revised August 16, 2020; accepted August 28, 2020. Date of publication September 11, 2020; date of current version May 3, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant U1736212, Grant U19A2068, Grant 61302161, and Grant 61972269, in part by the China Postdoctoral Science Foundation under Grant 2019TQ0217 and Grant 2020M673277, in part by the Provincial Key Research and Development Program of Sichuan under Grant 2020YFG0133, in part by the Fundamental Research Funds for the Central Universities under Grant YJ201933, in part by Doctoral Fund, Ministry of Education, China under Grant 20130181120076, and in part by the China International Postdoctoral Exchange Fellowship Program (Talent-Introduction). Paper no. TII-20-3189. *(Corresponding author: Liang Zhao.)*

Beibei Li, Yuhao Wu, Jiarui Song, Tao Li, and Liang Zhao are with the College of Cybersecurity, Sichuan University, Chengdu 610065, China (e-mail: libeibei@scu.edu.cn; wuyuhao@stu.scu.edu.cn; jiaruisong@stu.scu.edu.cn; bli012@e.ntu.edu.sg; zhaoliangjapan@scu.edu.cn).

Rongxing Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: RLU1@unb.ca).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2020.3023430

I. Introduction

INDUSTRIAL cyber–physical systems (CPSs) are generally referred to as large-scale, geographically-dispersed, complex, and heterogeneous Internet-of-Things (IoT) in an industrial context, such as smart grids, autonomous transportation systems, and gas pipelining systems [1]–[3]. Industrial CPSs are encapsuled with intelligent networking and computing technologies, such as 5G (and beyond), software-defined networking (SDN), network function virtualization, cloud computing, and artificial intelligence (AI), with existing industrial control systems (ICSs), a general architecture of which is shown in Fig. 1. Industrial CPSs are envisioned to facilitate remote access, promote smart services, enable big data analytics, and allow better provisioning of network resources [4].

The benefits from industrial CPSs seem clear, but these advancements have not come without risk [5]–[7]. Legacy industrial infrastructures have been implemented with poor security measures, leaving numerous potential vulnerabilities unattended. The rapid fusion of advanced networking and computing technologies has dramatically expanded the threat landscape by opening up new vulnerabilities that can be exploited across softwarized endpoints, networks, applications, and cloud services. One high-profile security incident is the BlackEnergy malware-based cyber assault on Ukraine's power grid in December 2015 [8], where more than 30 power substations were switched OFF, and about 230 thousand people were left in dark for a period from one to six hours. Other notorious cyber incidents associated with industrial CPSs include the Stuxnet on Iran's nuclear power plant [9], VPNFilter on supervisory control and data acquisition (SCADA) protocols [10], unauthorized penetration on Australia's Maroochy sewage factory [11], etc. Such incidents demonstrate that industrial CPSs are much likely to remain ongoing targets of interest in the near future, particularly by state-sponsored or affiliated actors. The importance of cybersecurity in industrial CPSs are reinforced by the U.S. Department of Homeland Security in the 2016 ICS-CERT Annual Assessment Report [12], which remarked that *"rapid increases in the connectivity of operational technology through the Internet of Things raises new challenges for control systems security,"* and also by the U.S. Department of Commerce in the NIST Guide to ICS security [13], stating that "*cybersecurity is essential to the safe and reliable operation of modern industrial processes.*"

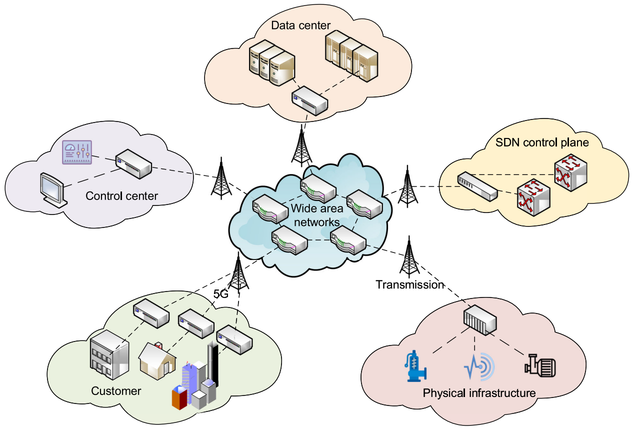


Fig. 1. General architecture of industrial CPSs.

The state-of-the-art literature has seen an increasing interest in addressing cybersecurity issues pertaining to industrial CPSs, placing priorities on AI relevant intrusion detection schemes in recent years. For example, in 2019, Qiu *et al.* [14] developed a dueling deep Q-learning-based approach to mitigating cyber threats against safe communications in software-defined industrial IoT. In early 2020, Ismail *et al.* [15] investigated electricity theft attacks in smart grid CPSs, and proposed a deep learning-based intrusion detection system for such cyberattacks. More recent studies can be seen in Section II. Unfortunately, most of the existing AI relevant intrusion detection schemes associated with industrial CPSs are developed on a strong assumption that sufficient high-quality examples of cyberattacks on industrial CPSs are always available, readily for the system defender to build a desired intrusion detection model. In real-world scenarios, however, one industrial CPS owner usually has rather limited attack examples, making the model building work incredibly challenging. Further, industrial CPS owners are usually unwilling to share such attack examples (neither those normal behavior examples) to the third parties, because highly-sensitive information about their critical industrial CPSs is always involved in these data resources. In such situations, we see that building a desired AI-based intrusion detection model for industrial CPSs is an apparently intractable task. Specifically, we first design a novel deep learning model, based on CNN and gated recurrent unit (GRU), to detect various types of cyber threats against industrial CPSs. In addition, we develop a new federated learning framework for multiple industrial CPS owners to collectively build a comprehensive intrusion detection model in a privacy-preserving way. Moreover, we design a secure communication protocol based on the Paillier public-key cryptosystem to preserve the security and privacy of model parameters through the training process. The main contributions of this work are threefold.

1) First, we create a novel deep learning-based intrusion detection model for industrial CPSs, by making use of CNN and GRU. This model is highly effective in detecting various types of cyber threats against industrial CPSs, such as denial-of-service (DoS), reconnaissance, response injection, and command injection attacks.
2) Second, a federated learning framework is developed, which, on the one hand, enables building a comprehensive intrusion detection model by taking advantage of data resources from multiple industrial CPS owners (in the same domain). On the other hand, this framework supports data processing at each industrial CPS's own premise, allowing effective privacy preservation of data resources.
3) Third, we craft a Paillier public-key cryptosystem-based secure communication protocol for the developed federated learning framework, by which the security and privacy of model parameters through the training process can be well preserved.

The remainder of this article is organized as follows. In Section II, we review the state-of-the-art studies on intrusion detection schemes for industrial CPSs and federated learning-based intrusion detection methods. In Section III, we introduce the system model and threat model considered in this work. Section IV elaborates on the proposed DeepFed scheme. Section VI gives the performance evaluation. Finally, Section VI concludes this article.

II. Related Work

In this section, we briefly review the state-of-the-art studies focusing on intrusion detection schemes for industrial CPSs as well as federated learning-based intrusion detection methods.

A. Intrusion Detection Schemes for Industrial CPSs

Recent years have witnessed an increasing research interest in intrusion detection schemes in the context of industrial CPSs. For example, in 2018, Yang *et al.* [16] designed an approach based on zone partition to detect both known and unknown cyberattacks for industrial CPSs, even when several zones are compromised simultaneously. Also, Wang *et al.* [17] in 2018 proposed a stacked auto-encoder-based deep learning scheme to detect two-stage sparse cyberattacks against the ac state estimation in smart grid CPSs. In 2019, Qiu *et al.* [14] developed a dueling deep Q-learning-based approach to mitigate cyber threats against safe communications in software-defined industrial IoT. In the same year, Yang *et al.* [18] designed a convolutional neural network (CNN)-based intrusion detection system for SCADA networks, in order to protect industrial CPSs from both conventional and SCADA specific network-based cyberattacks. In early 2020, Ismail *et al.* [15] investigated electricity theft attacks in smart grid CPSs and proposed a deep learning-based intrusion detection system for such cyberattacks. Also in 2020, Liu *et al.* [19] presented a hierarchically distributed intrusion detection framework for the security monitoring of large-scale industrial CPSs. It takes advantage of the security monitoring of physical systems and information systems to achieve all-round security protection of industrial CPSs.

B. Federated Learning-Based Intrusion Detection Methods

Emerged as a promising tool for addressing data islands issues in recent years, federated learning has been widely adopted

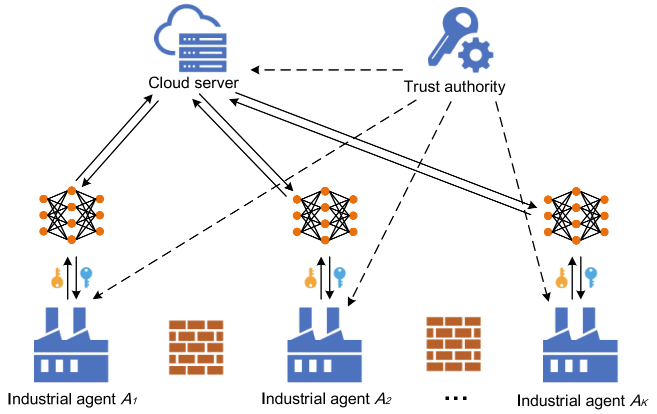


Fig. 2. System model under consideration.

in many areas. Particularly, a series of researchers have recently conducted federated learning-based studies to achieve intrusion detection. For example, in 2018, Preuveneers *et al.* [20] described a permissioned blockchain-based federated learning method to achieve an anomaly detection machine learning model, where contributing parties in federated learning can be held accountable and have their model updates audited. In 2019, Nguyen *et al.* [21] designed an autonomous self-learning distributed system for detecting compromised IoT devices, which employed a federated learning approach to achieve intrusion detection. In the same year, Zhao *et al.* [22] proposed a multitask deep neural network in federated learning (MT-DNN-FL) to perform network anomaly detection task. In 2020, Chen *et al.* [23] proposed a federated deep autoencoding Gaussian mixture model (FDAGMM) to improve the disappointing performance of traditional DAGMM in network anomaly detection caused by limited data amount.

III. System Model and Threat Model

In this section, we introduce the system model and threat model considered in this work.

A. System Model

The system model under consideration is a federated deep learning framework (see Fig. 2), which mainly comprises three types of entities, i.e., a trust authority, a cloud server, and K industrial agents.

1) *Trust Authority*: The trust authority undertakes the task of bootstrapping the whole system, generating public keys and private keys for the Paillier public-key cryptosystem-based secure communication protocol, as well as establishing secure communication channels for the cloud server and each industrial agent.
2) *Cloud Server*: The cloud server is responsible for building a comprehensive intrusion detection model, by federating the model parameters of those locally learned at each industrial agent's own premise. Multiple rounds of interactions between the cloud server and each industrial agent are demanded in order to obtain a final "perfect" intrusion detection model.
3) *Industrial Agents*: Each industrial agent, on behalf of the industrial CPS owner, is in charge of building a local intrusion detection model based on its own collected industrial CPS data and aiding in updating the parameters of the intrusion detection model by recurrently interacting with the cloud server.

B. Threat Model

In the threat model, we consider cyber threats both targeting the industrial CPSs and those aiming at our federated deep learning framework.

1) Cyber Threats Against Industrial CPSs: Unlike traditional computer systems, industrial CPSs are being exposed to not only traditional cyber threats, such as DoS and DDoS attacks, but also a line of highly customized new cyber threats tailored to industrial systems, such as command injection and response injection attacks. In this article, we consider all the abovementioned cyber threats, with a focus on the following.

a) *Reconnaissance attacks* are usually conducted for gathering valuable information about industrial CPSs, mapping the network architectures, and identifying device features, such as the manufacturer, model number, supported network protocols, and device addresses.
b) *Response injection attacks* are generally carried out to interfere with monitoring and reporting the state of a remote process in industrial CPSs. These attacks can falsify responses reporting to querying parties, such that biased system state information is provided.
c) *Command injection attacks* are launched often by injecting falsified control or configuration commands to mislead system behaviors of industrial CPSs. Such attacks can cause unauthorized modification of device configurations, process setpoints, or communication destinations.
d) *DoS attacks* are mounted usually by flooding the targets with superfluous requests in an extremely high frequency to exhaust the resources of server systems in industrial CPSs, which can disrupt the services or prevent legitimate requests from being fulfilled.

2) Cyber Threats Against Federated Learning Framework: In the considered federated deep learning framework, it is assumed that the trust authority is a fully trusted party, and the cloud server is a semihonest party who is honest in conducting all the given tasks but curious about the model parameters of the intrusion detection model. Also, we assume that all industrial agents are semihonest, who strictly follow the designed protocols but may be interested in other agents' data resources. Further, it is also taken into consideration that malicious eavesdroppers or other external attackers may intercept with the communication links in an attempt to access both data resources of each industrial CPSs and the parameters of the intrusion detection model. In this case, we consider the following two types of cyber threats.

a) *Eavesdropping of data resources*: As for the industrial CPS owners, their data resources for training the intrusion detection model, particularly for those attack examples, are highly sensitive and even national critical. If shared with the cloud server, it may lead to considerable business losses or severe national security risks.

b) *Eavesdropping of model parameters*: The parameters of an intrusion detection model contain critical information about the data resources. If they are accessed by the outside world in an unauthorized way, some basic knowledge of such data resources, e.g., type of cyber threats or its example distributions may possibly be leaked.

IV. Proposed DeepFed Scheme

In this section, we elaborate on the proposed DeepFed scheme by outlining the scheme workflow first, and then introducing the designed CNN-GRU-based intrusion detection model, followed by the Paillier-based secure communication protocol.

A. Workflow of the DeepFed Scheme

The basic idea of the DeepFed scheme is networking multiple industrial CPS owners to collectively build a deep learning intrusion detection model, based on a developed federated learning framework along with a Paillier-based secure communication protocol. The complete workflow of the DeepFed scheme can be described in five phases, which is given below (see also Algorithm 1 for the workflow).

1) System Initialization: In the system initialization phase, the trust authority bootstraps the whole system by conducting $KeyGenerate(\kappa)$ (see more details in Section IV-C), by which the public key $\mathcal{PK} = \{n, g\}$ as well as the private key $\mathcal{SK} = \{\lambda, \mu\}$ used in the Paillier-based secure communication protocol can be generated, and a secure channel between the cloud server and each industrial agent is established. Then, the cloud server selects an array of initial parameters $\mathbf{w}^0$ for the deep learning-based intrusion detection model and some other parameters relevant to the model training, i.e., the learning rate η, exponential decay rates for moment estimates ρ_1, ρ_2 $\in [0, 1)$, a small constant used for numerical stabilization ς, loss function $\mathfrak{L}$, and batch size B. In addition, each industrial agent A_k reports the size N_k of its own data resource $\mathcal{D}_k$, to the cloud server, where $k \in \mathcal{K} = \{1, 2, \cdots, K\}$, and then, the cloud server computes a contribution ratio for each industrial agent by $\alpha_k = N_k / \ (N_1 + N_2 + \cdots + N_K)$. Last, define a positive integer R denoting the total rounds of communications between the cloud server and an industrial agent.

2) Local Model Training by Industrial Agents: After receiving initial model parameters $\mathbf{w}^0$ as well as $\eta, \rho_1, \rho_2, \varsigma, \mathfrak{L}, B$ from the cloud server, each industrial agent trains a deep learning-based intrusion detection model locally, using their own private data resource $\mathcal{D}_k$ $(k \in \mathcal{K})$. The detailed training procedure is summarized in Algorithm 2. Since the local model training is performed offline, it is assumed that sufficiently strong computational capabilities can be provided, so that there is no need to care too much about the computational complexity of this algorithm.

3) Model Parameters Encryption by Industrial Agents: When a local deep learning model is trained, each industrial agent A_k encrypts the model parameters $\mathbf{w}_k^r$ using $ParaEncrypt(w_{k,j}^r, \mathcal{PK})$, where $\mathbf{w}_k^r = (w_{k,1}^r, w_{k,2}^r, \cdots, w_{k,T}^r)$ and $j \in \mathcal{T} = \{1, 2, \cdots, T\}$. Then, the encrypted parameters $\{E_{Pai}(w_{k,j}^r) | j \in \mathcal{T}\}$ of the local deep learning model are then uploaded to the cloud server by each industrial agent, where T is the total number of parameters in a local deep learning model.

```
Algorithm 1: Privacy-Preserving Federated Learning
Input: The security parameter $\kappa$, industrial agents set $\mathcal{A}$,
       data resources of all industrial agents $\{\mathcal{D}_k |$
       $k \in \mathcal{K}\}$, number of communication rounds $R$.
Output: The comprehensive deep learning model.
Initialization:
 a). The trust authority generates the key pair by
 $\{\mathcal{PK}, \mathcal{SK}\} = KeyGenerate(\kappa)$;
 b). The trust authority establishes a secure channel for
 the cloud server and each industrial agent;
 c). The cloud server initializes $\eta, \rho_1, \rho_2, \varsigma, \mathfrak{L}, B$, and
 initial model parameters $\mathbf{w}^0$;
 d). Each $A_k$ reports a size $N_k$ to the cloud server, where
 $k \in \mathcal{K}$; then, the cloud server computes each contribution
 ratio by $\alpha_k = N_k/(N_1 + N_2 + ... + N_K)$;
 e). Initialize the communication round index by $r = 1$.
Procedure:
for $r \leq R$ do
    (I). For industrial agents:
    for $\forall k \in \mathcal{K}$ do
        $A_k$ computes the $r$-th round local model
        parameters $\mathbf{w}_k^r$ as per Algorithm 2 with inputs: $\eta$,
        $\rho_1$, $\rho_2$, $\varsigma$, $\mathfrak{L}$, $B$, $\mathbf{w}^{r-1}$, $\mathcal{A}$, $\mathcal{D}_k$;
        for $\forall j \in \mathcal{T}$ do
            $E_{Pai}(w_{k,j}^r) = ParaEncrypt(w_{k,j}^r, \mathcal{PK})$;
        end
        $A_k$ uploads the encrypted model parameters
        $\{E_{Pai}(w_{k,j}^r) | j \in \mathcal{T}\}$ to the cloud server;
    end
    (II). For cloud server:
    for $\forall j \in \mathcal{T}$ do
        $c_j$ = $ParaAggregate(w_{1,j}^r, \cdots, w_{K,j}^r,$
        $\alpha_1, \cdots, \alpha_K)$;
    end
    The cloud server distributes the aggregated
    ciphertexts $\mathbf{c} = \{c_j | j \in \mathcal{T}\}$ to all $A_k (k \in \mathcal{K})$;
    (III). For industrial agents:
    for $\forall k \in \mathcal{K}$ do
        for $\forall j \in \mathcal{T}$ do
            $\tilde{w}_{k,j}^r = ParaDecrypt(c_j, \mathcal{SK})$;
        end
        $A_k$ updates its local deep learning model using
        the updated parameters $\tilde{\mathbf{w}}^r = \{\tilde{w}_{k,j}^r | j \in \mathcal{T}\}$;
    end
    $r \leftarrow r + 1$.
end
return The comprehensive deep learning model with
parameters $\mathbf{w}^R$.
```

4) Model Parameters Aggregation by the Cloud Server: Given the contribution ratios and encrypted model parameters from all industrial agents, the cloud server aggregates them by $ParaAggregate(E_{Pai}(w_{k,1}^r), \cdots, E_{Pai}(w_{k,T}^r), \alpha_1, \cdots, \alpha_K)$. Then, the aggregated ciphertexts $\mathbf{c} = \{c_j | j \in \mathcal{T}\}$ are sent back to the industrial agents.

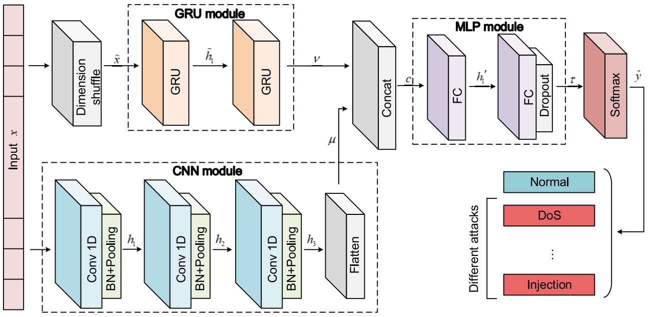


Fig. 3. Architecture of designed CNN-GRU model.

5) Local Model Updating by Industrial Agents: By decrypting the ciphertexts **c** using $ParaDecrypt(c_j, \mathcal{SK}(j \in \mathcal{T})$, each industrial agent obtains the updated model parameters $\tilde{\mathbf{w}}^r$. Then, the parameters of the local deep learning model are then updated by $\tilde{\mathbf{w}}^r$.

After R rounds (an empirically determined threshold) of interactions between the cloud server and industrial agents, a comprehensive deep learning-based intrusion detection model can finally be obtained. As we can see from Algorithm 1, each industrial agent A_k needs to conduct parameter encryption and decryption tasks, which in total require T exponentiation operations in $\mathbb{Z}^*_{n^2}$ and a line of multiplication operations in $\mathbb{Z}^*_{n^2}$ (that can be relatively negligible) in each communication round. In this way, the computational cost of each industrial agent A_k is almost linearly proportionally to the total number of parameters T in a local deep learning model. As for the cloud server, it only needs to perform K times of multiplication operations in $\mathbb{Z}^*_{n^2}$ when aggregating all industrial agents' model parameters in each communication round.

B. CNN-GRU-Based Intrusion Detection Model

In this part, we introduce the newly designed CNN-GRU-based intrusion detection model.

1) Model Architecture: The designed model is mainly composed of a CNN module and a GRU module, followed by an multilayer perceptron (MLP) module, and then a softmax layer (see Fig. 3), they are respectively described as below:

a) *CNN Module*: The CNN module mainly involves three convolutional blocks, and each convolutional block consists of a convolutional layer, a batch normalization layer, and a max-pooling layer.

b) *GRU Module*: The GRU module is composed of two identical GRU layers.

c) *MLP Module*: The MLP module involves two fully connected layers and a dropout layer (used to prevent the model from overfitting).

d) *Softmax Layer*: The softmax layer is exploited to map the nonnormalized output of the MLP module to a probability distribution over predicted classes.

Given a feature vector x (a one-dimensional vector denoting the numerical features of a network traffic data example) being the input of the designed model, the GRU module and CNN module then process it, respectively. Specifically, as for the GRU module, it regards x as a multivariate time series with a single time step. Prior to delivering x to the GRU module, a dimension shuffle layer is implemented, which transposes the temporal dimension of the feature vector. It is given by $\tilde{x} = \text{Shuffle}(x)$. Then, the GRU module processes $\tilde{x}$ in the following ways with the purpose of extracting the temporal patterns:

$$\tilde{h_1} = \text{GRU}_1(\tilde{x}), \text{ and } \nu = \text{GRU}_2(\tilde{h_1}) \tag{1}$$

where $\text{GRU}_i,\ i \in \{1, 2\}$, represents the ith GRU layer, $\tilde{h_1}$ is a hidden vector, and ν is the final output of the GRU module.

When it comes to the CNN module, it treats x as a univariate time series with multiple time steps

$$\begin{aligned} h_1 &= \text{ConvBlock}_1(x) \\ h_2 &= \text{ConvBlock}_2\,(h_1) \\ h_3 &= \text{ConvBlock}_3\,(h_2) \\ \mu &= \text{Flatten}\,(h_3) \end{aligned} \tag{2}$$

where the $\text{ConvBlock}_i,\ i \in \{1, 2, 3\}$, represents the ith convolutional block in the CNN module, $h_1, h_2, h_3 \in \mathbb{R}^k$ are hidden vectors. Then, the output of the three convolutional block is transferred to a flatten layer to be flattened, the result of which is μ. Following the CNN module and GRU module, μ and ν are concatenated and then fed into the MLP module, which is described by

$$\begin{aligned} c &= \text{Concate}(\mu, \nu) \\ h_1' &= \text{FC}_1(c) \\ h_2' &= \text{FC}_2(h_1') \\ \tau &= \text{Dropout}(h_2') \end{aligned} \tag{3}$$

where Concate represents the concatenation operation, c is the concatenated result, FC_1 and FC_2 denotes the two fully connected layer, Dropout denotes the dropout layer. Moreover, h_2' and τ are the output of the two fully connected layer and the dropout layer, respectively. At last, the softmax layer provides the final classification result by $\hat{y} = \text{Softmax}(\tau)$, where Softmax represents the softmax layer and y is the final classification result of the network traffic data.

Since the CNN-GRU model performs multiclassification to detect Γ types of attacks in industrial CPSs, the cross-entropy function is used as the loss function, which is defined by

$$\mathfrak{L} = -\frac{1}{B} \sum_{i=0}^{B-1} \sum_{j=0}^{\Gamma-1} y_{i,j} \log \hat{y}_{i,j} \tag{4}$$

where B denotes the batch size, $y_{i,j}$ is the true label, and $\hat{y}_{i,j}$ is the probability that the ith example is predicted to be the jth label.

2) Local Model Training: Each industrial agent $A_k (k \in \mathcal{K})$ locally train the proposed deep learning model on their own data resource $\mathcal{D}_k$, with reference to Algorithm 2. Specifically, in the rth communication round, each industrial agent A_k first updates model parameters $\mathbf{w}_k^r$ based on the given updated model parameters $\tilde{\mathbf{w}}^r$. Then, using the same data resource $\mathcal{D}_k$, industrial agent A_k retrains the deep learning model based on the

Algorithm 2: Local Deep Learning Model Training

```
Input: $\eta, \rho_1, \rho_2, \varsigma, \mathfrak{L}, B$, $\mathbf{w}^{r-1}$, $\mathcal{A}$, $\mathcal{D}_k$
Output: $\mathbf{w}_k^r$
Initialization:
 a). Initialize the first and second moment variables by $s = 0$ and $v = 0$, respectively;
 b). Split $\mathcal{D}_k$ into batches with equal size $B$;
 c). Set the model parameters by $\mathbf{w}_k^r \leftarrow \tilde{\mathbf{w}}^{r-1}$;
Procedure:
repeat
    for each batch of data resource do
        a). Compute the gradient by $gd \leftarrow \bigtriangledown_{\mathbf{w}_k^r} \mathfrak{L}$;
        b). Update the biased first moment estimate by $s \leftarrow \rho_1 s + (1 - \rho_1) gd$;
        c). Update the biased second moment estimate by $v \leftarrow \rho_2 v + (1 - \rho_2) gd^2$;
        d). Compute the bias-corrected first moment estimate by $\hat{s} \leftarrow \frac{s}{1-\rho_1^e}$;
        e). Compute the bias-corrected second moment estimate by $\hat{v} \leftarrow \frac{v}{1-\rho_2^e}$;
        f). Update the model parameters by $\mathbf{w}_k^r \leftarrow \mathbf{w}_k^r - \eta \frac{\hat{s}}{\sqrt{\hat{v}+\varsigma}}$;
    end
until The loss function $\mathfrak{L}$ converages;
return $\mathbf{w}_k^r$
```

optimizer adaptive moment estimation (Adam) that is able to facilitate the convergence of the loss function.

C. Paillier-Based Secure Communication Protocol

In this part, we design a Paillier-based secure communication protocol for the developed federated learning framework. It is worth noting that the advanced encryption standard (AES) algorithm [24] is employed in our protocol to establish a secure channel between the cloud server and each industrial agent, which is helpful in mitigating malicious eavesdroppers and other external attackers. The Paillier cryptosystem [25], supporting an unlimited number of homomorphic additions, is exploited in our protocol to achieve secure and privacy-preserving federated learning over the cloud server. It is composed of the following four functions.

1) KeyGenerate(κ): Given a security parameter $\kappa \in \mathbb{Z}^+$, the trust authority generates the public key $\mathcal{PK} = (n, g)$ and the corresponding private key $\mathcal{SK} = (\lambda, \mu)$ as per the standard Paillier cryptosystem [25], where n is the product of two large prime numbers, $g \in \mathbb{Z}_{n^2}^*$ is a generator, $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$, and function L is defined as $L(\alpha) = (\alpha - 1)/n$. Then, the trust authority publishes the $\mathcal{PK}$ and distributes $\mathcal{SK} = (\lambda, \mu)$ to all the industrial agents. In addition, to establish a secure communication channel, the trust authority generates a symmetric key s_i for the cloud server and each industrial agent A_i, $i \in \{1, 2, \cdots, K\}$, respectively.

2) ParaEncrypt($m, \mathcal{PK}$): Define a function $\nu' = f(\nu) = 10^8 \cdot \nu \bmod n$, and given a message m, compute $m' = f(m)$. In this way, each model parameter is converted to a positive integer $m' \in \mathbb{Z}_n$. Select a random number $r \in \mathbb{Z}_n^*$ and encrypt the model parameter using the public key $\mathcal{PK}$ by

$$E_{Pai}(m) = g^{f(m)} \cdot r^n \bmod n^2 = g^{m'} \cdot r^n \bmod n^2. \tag{5}$$

3) ParaAggregate($E_{Pai}(m_1), \cdots, E_{Pai}(m_K), \alpha_1, \cdots, \alpha_K$): Given contribution ratios $\{\alpha_1, \alpha_2, \cdots, \alpha_K\}$ of each industrial agent, the cloud server amplifies these ratios by 1000 times to convert them as positive integers. With K model parameters $\{E_{Pai}(m_1), E_{Pai}(m_2), \cdots, E_{Pai}(m_K)\}$ in hand, the cloud server then aggregates these data by

$$\begin{aligned} c &= \prod_{i=1}^{K} E_{Pai}^{\alpha_i}(m_i) \\ &= g^{\alpha_1 m'_1} r_1^{\alpha_1 n} \cdot g^{\alpha_2 m'_2} r_2^{\alpha_2 n} \cdot g^{\alpha_K m'_K} r_K^{\alpha_K n} \bmod n^2 \\ &= g^{\sum_{i=1}^{K} \alpha_i m'_i} \cdot \prod_{i=1}^{K} r_i^{\alpha_i n} \bmod n^2. \end{aligned} \tag{6}$$

4) ParaDecrypt($c, \mathcal{SK}$): When receiving the ciphertext c of a summed updated model parameter from the cloud server, each industrial agent decrypts the summed updated model parameter $\tilde{m}'_{\text{sum}}$ by

$$\begin{aligned} \tilde{m}'_{\text{sum}} &= L(c \bmod n^2) \cdot \mu \bmod n \\ &= \frac{L(g^{\sum_{i=1}^{K} \alpha_i m'_i} \cdot \prod_{i=1}^{K} r_i^{\alpha_i n} \bmod n^2)}{L(g^\lambda \bmod n^2)} \bmod n \\ &= \sum_{i=1}^{K} \alpha_i m'_i \bmod n. \end{aligned} \tag{7}$$

Then, compute the average value of the summed updated model parameter by $\tilde{m}' = \tilde{m}'_{\text{sum}}/1000$. Recall that 1000 denotes a scalar used to convert the contribution ratios to a positive integer. Define a function $\nu = f^{-1}(\nu') = 10^{-8} \cdot \nu' \bmod n$. Considering that the original model parameters can either be positive (less than $n/2$ after the conversion by $\nu = f(\nu)$) or negative (larger than $n/2$ after the conversion), we recover the updated model parameter to the original scale by

$$\begin{cases} \tilde{m} = f^{-1}(\tilde{m}'), & \text{if } \tilde{m}' < \frac{n}{2}, \\ \tilde{m} = f^{-1}(\tilde{m}' - n), & \text{otherwise.} \end{cases} \tag{8}$$

V. Performance Evaluation

In this section, we conduct extensive experiments to evaluate the performance of our proposed DeepFed scheme. First, we give the experiment settings, including the environmental setup, data resource description and partitioning, baseline studies, and performance metrics. Then, we carry out a series of experiments to compare the performance of our proposed intrusion detection model with some state-of-the-art studies, including the Schneble's [26], Nguyen's [21], and Chen's [27], under our developed federated learning framework. In addition, we also compare the performance of the developed intrusion detection model with those local intrusion detection models built by each industrial agent as well as the ideal intrusion detection model built by a central entity on all data resources.

TABLE I
NUMERICAL RESULTS OF INTRUSION DETECTION MODELS WITH VARYING COMMUNICATION ROUNDS UNDER THREE DIFFERENT SCENARIOS

K	R	Schneble et al. [26]				Nguyen et al. [21]				Chen et al. [27]				The Proposed DeepFed			
		Accuracy	*Precision*	*Recall*	*F-score*	*Accuracy*	*Precision*	*Recall*	*F-score*	*Accuracy*	*Precision*	*Recall*	*F-score*	*Accuracy*	*Precision*	*Recall*	*F-score*
3	2	0.9812	0.9870	0.9639	0.9748	0.9818	0.9882	0.9623	0.9746	0.9843	0.9860	0.9674	0.9762	0.9915	0.9922	0.9690	0.9802
	4	0.9816	0.9879	0.9645	0.9756	0.9903	0.9883	0.9676	0.9776	0.9906	0.9928	0.9680	0.9799	0.9919	0.9938	0.9693	0.9810
	6	0.9818	0.9880	0.9646	0.9757	0.9906	0.9884	0.9676	0.9776	0.9911	0.9932	0.9683	0.9803	0.9920	0.9938	0.9690	0.9809
	8	0.9818	0.9880	0.9646	0.9757	0.9908	0.9886	0.9676	0.9777	0.9912	0.9933	0.9684	0.9803	0.9919	0.9938	0.9680	0.9804
	10	0.9820	0.9882	0.9646	0.9759	0.9909	0.9886	0.9676	0.9778	0.9913	0.9934	0.9682	0.9803	0.9920	0.9886	0.9736	0.9810
5	2	0.9806	0.9861	0.9614	0.9731	0.9811	0.9869	0.9640	0.9749	0.9840	0.9882	0.9628	0.9748	0.9909	0.9918	0.9682	0.9796
	4	0.9814	0.9873	0.9644	0.9753	0.9900	0.9920	0.9628	0.9768	0.9906	0.9928	0.9680	0.9799	0.9915	0.9871	0.9742	0.9805
	6	0.9816	0.9875	0.9645	0.9754	0.9905	0.9923	0.9627	0.9769	0.9912	0.9932	0.9684	0.9803	0.9919	0.9937	0.9691	0.9809
	8	0.9816	0.9874	0.9645	0.9754	0.9907	0.9923	0.9629	0.9771	0.9913	0.9926	0.9680	0.9798	0.9919	0.9881	0.9744	0.9811
	10	0.9816	0.9878	0.9645	0.9756	0.9909	0.9924	0.9645	0.9779	0.9913	0.9934	0.9684	0.9804	0.9920	0.9885	0.9745	0.9813
7	2	0.9807	0.9866	0.9627	0.9740	0.9815	0.9875	0.9645	0.9755	0.9837	0.9885	0.9639	0.9756	0.9847	0.9865	0.9678	0.9767
	4	0.9811	0.9873	0.9638	0.9750	0.9902	0.9933	0.9631	0.9776	0.9905	0.9925	0.9640	0.9777	0.9918	0.9937	0.9685	0.9806
	6	0.9815	0.9874	0.9644	0.9754	0.9903	0.9925	0.9632	0.9773	0.9911	0.9928	0.9679	0.9798	0.9919	0.9937	0.9686	0.9807
	8	0.9816	0.9874	0.9645	0.9754	0.9906	0.9927	0.9633	0.9775	0.9912	0.9901	0.9682	0.9787	0.9920	0.9886	0.9734	0.9808
	10	0.9817	0.9879	0.9645	0.9757	0.9909	0.9931	0.9634	0.9777	0.9913	0.9903	0.9685	0.9790	0.9920	0.9885	0.9747	0.9814

A. Experiment Settings

1) Environmental Setup: The designed CNN-GRU model is implemented using the Keras API[1] and the federated learning framework is built by a lightweight Python framework Flask.[2] Our experiments are conducted on a Ubuntu 18.04.3 LTS platform with an Intel Xeon E5-2618L v3 CPU and an NVIDIA GeForce RTX 2080TI GPU (64GB RAM).

2) Data Resource Description and Partitioning: We conduct experiments on a real data resource of a gas pipelining system (one significant example of industrial CPSs) [28]. In this data resource, one class of network data under normal operations and seven classes under various cyberattacks are, respectively, collected. Each piece of network data in this data resource contains 26 features and 1 label. In our experiments, the data resource is divided into two major parts, i.e., 80% for training and 20% for testing, and the training part is further divided into even partitions to each industrial agent for local model training. Note that all the trained deep learning models are tested on the same testing data.

3) Baseline Studies: In this work, we compare the performance of our proposed DeepFed scheme with some state-of-the-art studies, where federated learning frameworks are also used. Schneble *et al.* [26] proposed a single layer MLP-based federated learning framework for attack detection in medical CPSs. Also, Nguyen *et al.* [21] presented a three-hidden-layer GRU-based federated self-learning system for intrusion detection in IoT networks. Further, Chen *et al.* [27] utilized a CNN-based federated framework for data classifications, which is composed of two convolutional layers, two max-pooling layers, two fully connected layers, and one softmax layer. We fully reproduce these deep learning models in our work and compare the performance with our designed model under the proposed federated learning framework.

4) Performance Metrics: Four common metrics are used to evaluate the performance of the detection model as follows.

a) Accuracy: The results of the model to predict the correct proportion.

b) Precision: The proportion of examples identified as cyberattacks that are indeed cyberattacks.

c) Recall: The proportion of all cyberattacks examples correctly identified as exact types of cyberattacks.

d) F-score: The weighted average of precision and recall. Note that, the macro averaged values are utilized to comprehensively evaluate the performance of all considered intrusion detection models.

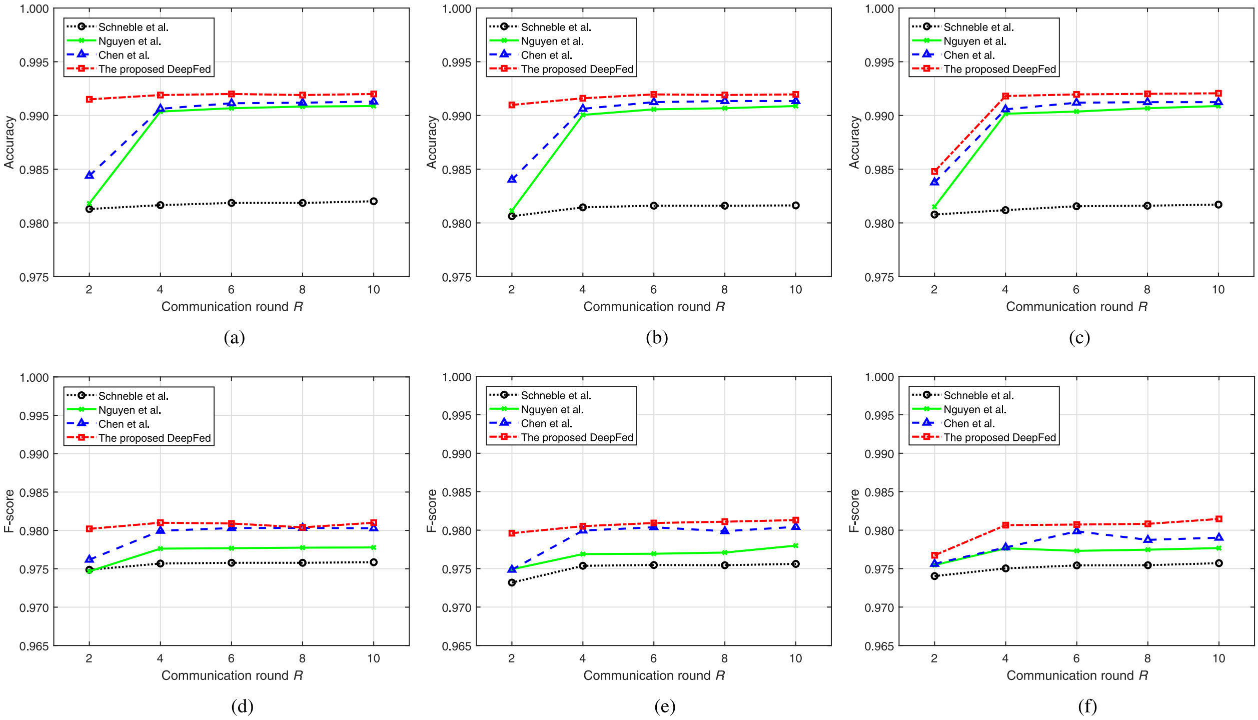
B. Performance Comparison with State-of-the-Art Studies

We first conduct experiments to compare the performance of our proposed DeepFed scheme with the abovementioned baseline studies [21], [26], [27]. Three groups of experiments are conducted, where different numbers of industrial agents $K = 3$, 5, and 7 are, respectively, considered.

Table I shows the numerical results about the performance of federated intrusion detection models, in terms of the accuracy, precision, recall, and F-score, under three different scenarios with $R = 2$, 4, 6, 8, and 10, respectively. It can be easily seen that, the proposed intrusion detection model outperforms other state-of-the-art studies on all metrics. As the number of communication rounds R increases from 1 to 10, the performance of each intrusion detection model generally improves, and gradually stabilizes when R is sufficiently large. It's worth noting that, we can obtain an accuracy, precision, recall, F-score of 99.20%, 98.86%, 97.34%, and 98.08%, respectively, when $K = 3$, 99.20%, 98.85%, 97.45%, and 98.13% when $K = 5$, and 99.20%, 98.85%, 97.47%, 98.14% when $K = 7$, respectively,

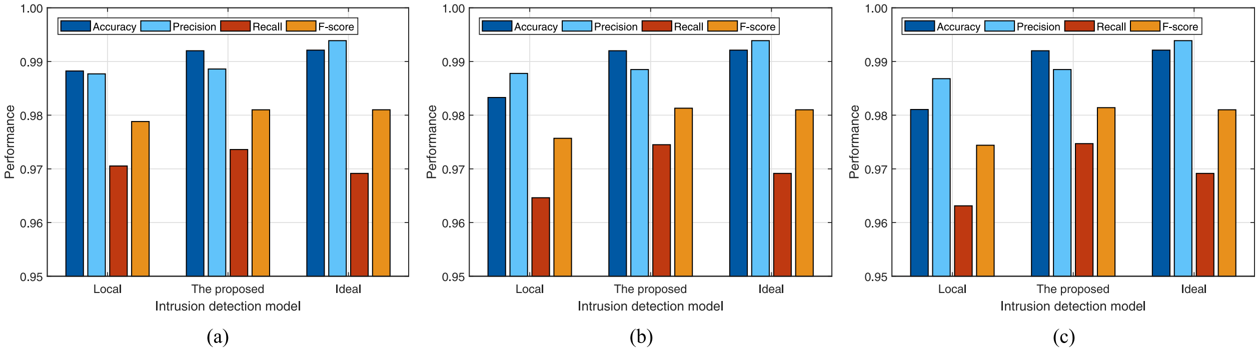
[1] Keras: Python deep learning library (http://keras.io/).

[2] Flask: Python web development framework (http://flask.pocoo.org/).



(a) (b) (c) (d) (e) (f)

Fig. 4. Comparison of the accuracy and F-score of considered intrusion detection models with varying communication rounds under three different scenarios.(a) Accuracy versus R ($K = 3$). (b) Accuracy versus R ($K = 5$). (c) Accuracy versus R ($K = 7$). (d) F-score versus R ($K = 3$). (e) F-score versus R ($K = 5$). (f) F-score versus R ($K = 7$).



(a) (b) (c)

Fig. 5. Performance comparison of the local, ideal, and the proposed intrusion detection models under three different scenarios. (a) $K = 3, R = 10$. (b) $K = 5, R = 10$. (c) $K = 7, R = 10$.

with the communication round $R = 10$. Fig. 4 also visually presents the numerical results of the accuracy and F-score of all considered intrusion detection models with varying communication rounds, under $K = 3, 5$, and 7, respectively. It is clear that all intrusion detection models tend to converge after sufficient rounds of communication with the cloud server. Importantly, the proposed intrusion detection model has generally the best performance over other baselines.

C. Performance Comparison With Local and Ideal Models

In addition to the above experiments, we also carry out experiments to evaluate the performance of each locally built intrusion detection model using limited data resources as well as the performance of an ideal model built by a central entity using all the data resources. Fig. 5 shows the numerical results of all four metrics under the abovementioned local, ideal, and the proposed intrusion detection models, respectively, with varying settings of K. As we can see, all the local intrusion detection models perform unsatisfactorily compared with the proposed model. Importantly, we also observe that the proposed model produces sufficiently good performance compared with the ideal model. It is, therefore, worth noting that the proposed model would be wise to all industrial CPS owners due to its high performance in intrusion detection and the ability to preserve the privacy of their data resources.

Furthermore, we also evaluate the performance of the local, ideal, and our proposed models in detecting various types of cyber threats against industrial CPSs. The numerical results are

TABLE II
NUMERICAL RESULTS OF THE LOCAL, IDEAL, AND PROPOSED MODELS IN DETECTING VARIOUS TYPES OF CYBER THREATS ($K = 5$)

Type of cyber threats	Local model			The proposed DeepFed			Ideal model		
	Precision	*Recall*	*F-score*	*Precision*	*Recall*	*F-score*	*Precision*	*Recall*	*F-score*
Naive malicious response injection attack	0.9909	0.9009	0.9438	0.9562	0.9476	0.9519	1.0000	0.9024	0.9487
Complex malicious response injection attack	0.9550	0.9838	0.9691	0.9904	0.9997	0.9950	0.9917	0.9997	0.9957
Malicious state command injection attack	0.9932	0.9359	0.9637	0.9932	0.9359	0.9637	0.9932	0.9359	0.9637
Malicious parameter command injection attack	0.9792	0.9856	0.9824	0.9792	0.9856	0.9824	0.9792	0.9856	0.9824
Malicious function command injection attack	1.0000	0.9478	0.9732	1.0000	0.9478	0.9732	1.0000	0.9478	0.9732
Denial-of-service attack	0.9955	0.9771	0.9862	0.9945	0.9864	0.9904	0.9945	0.9864	0.9904
Reconnaissance attack	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

summarized in Table II (taking $K = 5$ as an example). As can be seen that the proposed intrusion detection model exhibits excellent performance in terms of the precision, recall, and F-score while detecting multiple types of cyber threats against industrial CPSs, compared to a local model, and almost the same performance compared to an ideal model.

VI. CONCLUSION

In this article, we proposed a federated deep learning scheme, named DeepFed, for detecting and mitigating cyber threats against industrial CPSs. First, we developed a novel federated learning framework for multiple industrial CPSs, enabling the collective building of a comprehensive intrusion detection model in a privacy-preserving way. In addition, we created a novel CNN-GRU-based intrusion detection model, which allows effective detection of various types of cyber threats against industrial CPSs. Further, a Paillier-based secure communication protocol was designed for the federated learning framework, which effectively preserves the security and privacy of model parameters in the training process. Extensive experiments on a real industrial CPS dataset demonstrated the high effectiveness of the proposed DeepFed scheme as well as the superiorities over state-of-the-art schemes.

It is worth noting that the proposed scheme builds a federated intrusion detection model mainly for same-domain industrial CPSs. Future research directions will focus on addressing cybersecurity issues by federating data resources from different-domain industrial CPSs.

REFERENCES

[1] C. Lu, *et al.*, "Real-time wireless sensor-actuator networks for industrial cyber-physical systems," in *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1013–1024, May 2016.

[2] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4177–4186, Jun. 2020.

[3] B. Li, R. Lu, W. Wang, and K.-K. R. Choo, "Distributed host-based collaborative detection for false data injection attacks in smart grid cyber-physical system," *J. Parallel Distrib. Comput.*, vol. 103, pp. 32–41, May 2017.

[4] C. Chen, J. Yan, N. Lu, Y. Wang, X. Yang, and X. Guan, "Ubiquitous monitoring for industrial cyber-physical systems over relay-assisted wireless sensor networks," *IEEE Trans. Emerg. Topics Comput.*, vol. 3, no. 3, pp. 352–362, Sep. 2015.

[5] H. Bao, R. Lu, B. Li, and R. Deng, "BLITHE: Behavior rule based insider threat detection for smart grid," *IEEE Internet Things J.*, vol. 3, no. 2, pp. 190–205, Apr. 2016.

[6] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Trans. Ind. Informat.*, vol. 16, no. 10, pp. 6532–6542, Oct. 2019.

[7] B. Li, R. Lu, W. Wang, and K. R. Choo, "DDOA: A Dirichlet-based detection scheme for opportunistic attacks in smart grid cyber-physical system," *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 11, pp. 2415–2425, Nov. 2016.

[8] K. Zetter, "Inside the cunning, unprecedented hack of Ukraine's power grid," *Wired*, Mar. 2016. [Online]. Available: https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid/

[9] N. Falliere, L. O. Murchu, and E. Chien, "W32. Stuxnet dossier," Symantec Corp., Tempe, AZ, USA, White paper, vol. 5, Feb. 2011.

[10] Z. Bederna and T. Szadeczky, "Cyber espionage through botnets," *Secur. J.*, vol. 33, no. 1, pp. 43–62, Mar. 2020.

[11] N. Sayfayn and S. Madnick, "Cybersafety analysis of the Maroochy Shire sewage spill," MIT Interdisciplinary Consortium for Improving Critical Infrastructure Cybersecurity, MIT Management Sloan School, Cambridge, MA, USA, Working Paper CISL 2017-09, vol. 9, May 2017.

[12] J. Felker and M. Edwards, "ICS-CERT Annual Assessment Report," Industrial Control Systems Cyber Emergency Response Team, 2017, vol. S508C. [Online]. Available: https://www.us-cert.gov/sites/default/files/Annual-Reports/FY2016-Industrial-Control-Systems-Assessment-Summary-Report-S508C.pdf

[13] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, "Guide to Industrial Control Systems (ICS) Security," U.S. Department of Commerce, Washington, D.C., USA, NIST-800-82 (R2), May 2015. [Online]. Available: https:// nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf

[14] C. Qiu, F. R. Yu, H. Yao, C. Jiang, F. Xu, and C. Zhao, "Blockchain-based software-defined industrial Internet of Things: A dueling deep Q -learning approach," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4627–4639, Jun. 2019.

[15] M. Ismail, M. F. Shaaban, M. Naidu, and E. Serpedin, "Deep learning detection of electricity theft cyber-attacks in renewable distributed generation," *IEEE Trans. Smart Grid*, vol. 11, no. 4, pp. 3428–3437, Jul. 2020.

[16] J. Yang, C. Zhou, S. Yang, H. Xu, and B. Hu, "Anomaly detection based on zone partition for security protection of industrial cyber-physical systems," *IEEE Trans. Ind. Electron.*, vol. 65, no. 5, pp. 4257–4267, May 2018.

[17] H. Wang, J. Ruan, G. Wang, B. Zhou, Y. Liu, X. Fu, and J. Peng, "Deep learning-based interval state estimation of AC smart grids against sparse cyber attacks," *IEEE Trans. Ind. Informat*, vol. 14, no. 11, pp. 4766–4778, Nov. 2018.

[18] H. Yang, L. Cheng, and M. C. Chuah, "Deep-learning-based network intrusion detection for SCADA systems," in *Proc. IEEE Conf. Commun. Netw. Secur.*, Washington, DC, USA, Jun. 10–12, 2019, pp. 337–343.

[19] J. Liu, W. Zhang, T. Ma, Z. Tang, Y. Xie, W. Gui, and J. P. Niyoyita, "Toward security monitoring of industrial cyber-physical systems via hierarchically distributed intrusion detection," *Expert Syst. Appl.*, vol. 158, pp. 113 578–113 400, Nov. 2020.

[20] D. Preuveneers, V. Rimmer, I. Tsingenopoulos, J. Spooren, W. Joosen, and E. Ilie-Zudor, "Chained anomaly detection models for federated learning: An intrusion detection case study," *Appl. Sci.*, vol. 8, no. 12, pp. 2663–2683, Dec. 2018.

[21] 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 *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, Dallas, TX, USA, July7–10, 2019, pp. 756–767.

[22] Y. Zhao, J. Chen, D. Wu, J. Teng, and S. Yu, "Multi-task network anomaly detection using federated learning," in *Proc. 10th Int. Symp. Inf. Commun. Technol.*, Hanoi HaLong Bay, Vietnam, Dec.4–6, 2019, pp. 273–279.

[23] Y. Chen, J. Zhang, and C. K. Yeo, "Network anomaly detection using federated deep autoencoding Gaussian mixture model," in *Proc. Int. Conf. Mach. Learn. Netw.*, Paris, France, Dec.3–5, 2019, pp. 1–14.

[24] M. J. Dworkin *et al. Advanced Encryption Standard (AES)*, Nov. 2001, vol. 197. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

[25] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, Cologne, Germany, Apr. 26–30, 1999, pp. 223–238.

[26] W. Schneble and G. Thamilarasu, "Attack detection using federated learning in medical cyber-physical systems," in *Proc. Int. Conf. Comput. Commun. Netw.*, Valencia, Spain, Jul. 29–Aug. 1, 2019.

[27] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "FedHealth: A federated transfer learning framework for wearable healthcare," *IEEE Intell. Syst.*, vol. 35, no. 4, pp. 83–93, July–Aug. 2020.

[28] T. Morris and W. Gao, "Industrial control system traffic data sets for intrusion detection research," in *Proc. Int. Conf. Critical Infrastruct. Protection*, Arlington, TX, USA, Mar.17–19, 2014, pp. 65–78.

Beibei Li (Member, IEEE) received the B.E. (Hons.) degree in communication engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2014, and the Ph.D. degree in cybersecurity from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, in 2019.

He is currently an Associate Professor with the College of Cybersecurity, Sichuan University, Chengdu, China. He was invited as a Visiting Researcher with the Faculty of Computer Science, University of New Brunswick, Fredericton, Canada, from March to August 2018, and also the research group of Networked Sensing and Control, College of Control Science and Engineering, Zhejiang University, Hangzhou, China, from February to April 2019. His has authored or coauthored works in IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, *ACM Transactions on Cyber-Physical Systems*, IEEE INTERNET OF THINGS JOURNAL, *IFAC Automatica*, *Information Sciences*, IEEE ICC, and IEEE GLOBECOM, etc. His current research interests include several areas in security and privacy issues on cyber-physical systems (e.g., smart grids, industrial control systems, etc.), with a focus on intrusion detection techniques, artificial intelligence, and applied cryptography.

Dr. Li is serving or has served as a Publicity Chair, Publication Co-Chair, or a TPC member for several international conferences, including IEEE International Conference on Communications (ICC), IEEE Global Communications Conference (GLOBECOM), IEEE International Conference on Computing, Networking and Communications (ICNC), IEEE International Conference on Advanced Technologies for Communications (ATC), and International Conference on Wireless Communications and Signal Processing (WCSP).

Yuhao Wu is currently working toward the B.E. degree in cybersecurity with the College of Cybersecurity, Sichuan University, Chengdu, China.

He has authored or coauthored several papers in IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, *Knowledge-Based Systems*, and International Conference on Web Information Systems Engineering, etc. His research interests include cyber–physical system security, online social network security, and artificial intelligence.

Jiarui Song is currently working toward the B.E. degree in cybersecurity with the College of Cybersecurity, Sichuan University, Chengdu, China.

She has authored or coauthored works in IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS and IEEE Conference on Local Computer Networks. Her current research interests include intrusion detection, artificial intelligence, and social network analysis.

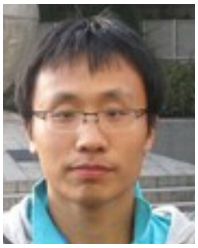
Rongxing Lu (Senior Member, IEEE) received the Ph.D. degree in cryptography from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012.

He is an Associate Professor with the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Fredericton, NB, Canada. Before that, he worked as an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, from April 2013 to August 2016. He worked as a Postdoctoral Fellow with the University of Waterloo, from May 2012 to April 2013.

Dr. Lu is a Senior Member of IEEE Communications Society. He was the recipient of the most prestigious Governor Generals Gold Medal, in 2012, the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013, and the 2016–17 Excellence in Teaching Award, FCS, UNB. He currently serves as the Vice-Chair (Conferences) of IEEE ComSoc CIS-TC.

Tao Li received the Ph.D. degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 1994.

He is currently a Professor with the College of Cybersecurity, Sichuan University, Chengdu, China. He has authored or coauthored nearly 300 papers in IEEE, ACM, Chinese Science, Science Bulletin, Natural Science Progress, and other important journals and academic conferences. His main research interests include immune computing, artificial immune systems, cloud computing, and cloud storage.



Liang Zhao (Member, IEEE) received the M.S. degree in computer science from Chongqing University, Chongqing, China, in 2009, and the Ph.D. degree in informatics from Kyushu University, Fukuoka, Japan, in 2012.

He is currently an Assistant Professor with the College of Cybersecurity, Sichuan University, Chengdu, China. He was a Visiting Researcher with the Surrey Center for Cyber Security of United Kingdom, from 2017–2018. His current research focuses on cryptography, in particular, provable security, verifiable (outsourced) computation, and postquantum cryptography.