GAN-Sec: Generative Adversarial Network Modeling for the Security Analysis of Cyber-Physical Production Systems

Sujit Rokka Chhetri, Anthony Bahadir Lopez, Jiang Wan, Mohammad Abdullah Al Faruque
{schhetri, anthl10, jiangwan, alfaruqu}@uci.edu
Department of Electrical Engineering and Computer Science
University of California, Irvine, California, USA

***Abstract*—Cyber-Physical Production Systems (CPPS) will usher a new era of smart manufacturing. However, CPPS will be vulnerable to cross-domain attacks due to the interactions between the cyber and physical domains. To address the challenges of modeling cross-domain security in CPPS, we are proposing GAN-Sec, a novel conditional Generative Adversarial Network based modeling approach to abstract and estimate the relations between the cyber and physical domains. Using GAN-Sec, we are able to determine if various security requirements such as confidentiality, availability, and integrity are met. We provide a security analysis of an additive manufacturing system to demonstrate the applicability of GAN-Sec.**

I. Introduction

The fourth industrial revolution will encompass a large scale use of Cyber-Physical Systems (CPS) known as Cyber-Physical Production Systems (CPPS) [1]. CPPS are an integration of sub-systems from multiple cyber and physical domains interconnected through communication networks. Using CPPS will aid in making the factory units smarter and adaptive; however, due to the tight interactions between the cyber and physical domains, there may exist various cross-domain vulnerabilities in CPPS.

There are two types of exploits of cross-domain vulnerabilities that we are primarily interested in: *kinetic-cyber attacks* [2] and *side-channel attacks*. *Kinetic-cyber attacks* refer to cyber-based attacks that directly impact the physical domain by breaching the integrity or availability of CPS [3]. *Side-channel attacks* refer to confidentiality attacks that steal critical information (from the cyber domain) by observing emissions from the physical domain [4], [5]. Both types of attacks may even be composed of smaller exploits that target confidentiality, integrity, or availability. Common examples of *kinetic-cyber attacks* include the notorious Stuxnet worm attack which caused physical damage to over 1,000 centrifuges in an Iranian nuclear plant [6], and a more recent cyber attack on a German Steel Mill, where attackers were able to cause massive damage to the blast furnace [7].

Existing CPPS modeling tools were created for the purpose of analyzing the system-level performance, reliability, energy efficiency, and quality of controls. Security is ignored by these tools and left as an afterthought of system design. However, most of the existing CPS security research work only target known vulnerabilities in specific domains. Afterward, they suggest an ad hoc repair such as patching software and/or replacing hardware components without showing that the repaired system is free of further vulnerabilities [8]. Some of these CPS also end up being a part of the overall CPPS.

Since these solutions cannot analyze and explore all the potential vulnerabilities, they can not answer the questions such as, "How secure is this CPPS against a confidentiality attack by a specific attacker?" or "Can we detect an integrity or availability attack on a CPPS?" To address these questions and improve the CPPS security research, we hypothesize that it is necessary to have a new modeling approach that takes into consideration the signal and energy flows of a system. We propose a security model that abstracts the relationship between the energy flows and signals flows to answer questions like the aforementioned ones.

A. Problem and Research Challenges

Notice that since the attack models mostly describe the capability of attackers rather than the system, one might assume that the existing attack models may be directly applied in CPPS security analysis. However, there exist the following research challenges to create a system-level model for CPS security analysis:

- The existing security properties are mostly only proposed for analyzing attacks in the cyber domain [8]. Thus, analyzing the cross-domain attacks in CPPS requires new types of security properties applicable in both the cyber and physical domains.
- The existing system-level behavior models in CPPS require various Models of Computation (MoCs)[1], for the cyber domain and physical domain (a MoC cannot be cross-domain) [9]. In order to analyze cross-domain security, a unified system behavior of interest for CPPS is required.
- In the CPPS environment, there are multiple sub-systems interacting with each other; therefore, information leakage or attack detection needs to be performed across multiple sub-systems.

To address these challenges, researchers have proposed to use information flow as the basis of a unified behavior model of cyber and physical domains for security analysis in [10]. However, [10] only focused on the application related to commodity flows, and proposed a coarse-grained information flow that models the commodity flow changes as an event. It fails to provide the capability of a quantified analysis of the cross-domain attack and detection capability.

[1]A MoC is a set of allowable operations used in computation and their respective costs (e.g., timing, performance, and memory overhead)

Modeling the information flows requires a statistical method. However, there is no guarantee that collected data during both design time and run-time is sufficient to create an accurate security model. In this work, we use generative adversarial networks (see Section I-B) to acquire a better estimate of the distribution (conditional in this paper) of the data. One hand if there is a large amount of data available, the discriminator of the generative adversarial network is able to estimate the data distribution, on the other, the generator, since it never sees the real data estimates the distribution without overfitting on the currently limited data, thus providing better distribution estimation. For security analysis, we consider abstracting a CPPS by its signal and energy flows and deriving the conditional densities among flow pairs using Conditional Generative Adversarial Network (CGAN).

B. Preliminaries

First, let us present some preliminary definitions about the energy and signal flows as well as GANs before explaining the details about our proposed CGAN-based security model of the CPPS.

Signal Flow: A signal flow is modeled as a discrete signal may be defined as a random variable F_S which have n number of possible values, $F_S \in \{f_1, f_2, ..., f_n\}$. We define a set of n events $E = \{E_1, E_2, ..., E_n\}$, where $E_i = 1$ when $F_S = f_i$, and $E_i = 0$ otherwise. We assume the probabilities for E_i are known as $Pr(E_i)$.

Energy Flow: We define energy flow as a continuous-domain time dependent variable F_E. Given a feature construction function $f_X(\cdot)$, we may construct a set of feature vectors $X = f_X(F_E)$. Next, given a feature extraction and selection function $f_Y(\cdot)$, we may extract a set of more relevant feature vectors $Y = f_Y(X)$. Assume $Y = \{Y^1, Y^2, ...Y^m\}$, where each feature vector Y^i may have n_i number of possible values $Y^i \in \{y^i_1, ..., y^i_{n_i}\}$. We define a set of n_i events $E_i = \{E_1, E_2, ..., E_{n_i}\}$, where any event E_{i_j} is true if $Y^i = y^i_j$ is met. We assume the probability for E_{i_j} is known as $Pr(E_{i_j})$.

Generative Adversarial Network: A Generative Adversarial Network (GAN) consists of a generative model G that captures the probability distribution of the flow's data $Pr(F_1)$, and a discriminator model D that estimates the probability that a sample of data x came from the training data rather than the generative model, $Pr(x \in F_1)$ [11]. GAN is a form of a maximum likelihood estimator that is trained to estimate the true probability distribution of a random variable by utilizing the empirical distribution derived from the training data. It achieves this by minimizing the *Kullback-Leibler divergence* between the empirical distribution and the generator's distribution, as follows [11]:

$$\theta^* = \arg\min_{\theta} D_{KL}(Pr_{data}(F_1) \mid\mid Pr_G(F_1; \theta)) \quad (1)$$

where θ are the model parameters. The models used in these networks are generally deep convolutional neural networks. Some advantages of using GAN over other methods is that it does not require Markov chains (where the chance of state explosion is high) and they can generate samples in parallel [11]. A Conditional Generative Adversarial Network (CGAN) is a variation of the GAN, where the generator and the discriminator both receive a conditional variable F_2 (see Figure 2). The primary objective of the generative model G is to learn the conditional probability distribution $Pr(F_1|F_2)$. In order to do this, it will require the labeled data (f_{1_1}, f_{2_1}), $(f_{1_2}, f_{2_2}), \ldots (f_{1_n}, f_{2_n})$. Here $\{f_{1_1}, f_{1_2}, \ldots, f_{1_n}\}$, $\{f_{2_1}, f_{2_2}, \ldots, f_{2_n}\}$ are the values taken by the random variable F_1 and F_2, respectively. G and D have a common objective function based on the two-player minimax game, as follows [12]:

$$\min_G \max_D V(D,G) = \mathbb{E}_{F_1 \backsim Pr_{data}(F_1)}[log(D(F_1|F_2))] + \mathbb{E}_{Z \backsim Pr_Z(Z)}[log(1 - D(G(Z|F_2)))] \quad (2)$$

where Z is a noise random variable that is provided along with F_1 and F_2 to G for training.

In Equation 2, the model G is trained to minimize $log(1 - D(G(Z|F_2)))$ so that it may generate samples more similar to the training data. D is trained to maximize $log(D(F_1|F_2))$ so that it may accurately differentiate between the real data and the generated data. Given enough time and capacity, the end result will be that G is trained to accurately estimate the $Pr(F_1|F_2)$ and D cannot differentiate G's output from real or generated data. This is an important concept for our security model because using $Pr(F_1|F_2)$, one can estimate a $(signal/energy)$ flow based on another flow.

C. Our Novel Contributions

To resolve the challenges of CPPS security modeling, our novel contribution are as follows:

- **Conditional Generative Adversarial Network-based Model for Security Analysis of CPPS (Section II)** which models the conditional probability distribution between the various information flows (cyber-domain data and physical domain energy flows).
- **Automatic Model Generation Algorithm (Section III)** which includes a graph search and pruning algorithm to reduce the complexity of the model and simulation/ experimental methods to estimate and generate the proposed behavior model.
- **A Case Study (Section IV)** for analyzing information leakage from multiple physical emissions in a single sub-system (additive manufacturing) to demonstrate the applicability of the proposed modeling technique in security analysis.

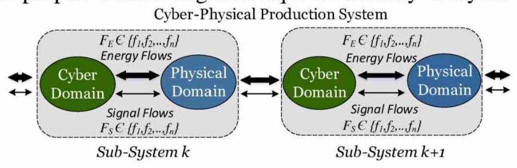

Cyber-Physical Production System
$F_E \in \{f_1, f_2 ... f_n\}$
Energy Flows
Cyber Domain
Physical Domain
Signal Flows
$F_S \in \{f_1, f_2 ... f_n\}$
Sub-System k
$F_E \in \{f_1, f_2 ... f_n\}$
Energy Flows
Cyber Domain
Physical Domain
Signal Flows
$F_S \in \{f_1, f_2 ... f_n\}$
Sub-System k+1

Fig. 1. *Cyber-Physical Production System (CPPS) with multiple sub-systems.*

II. CGAN-based CPPS Security Model

A typical CPPS model (see Figure 1) includes multiple sub-systems $(1, 2, \ldots, n)$. In each sub-system, there are cyber and physical domain components with energy and signal flows connecting them. Moreover, the energy and signal flows may occur among sub-systems as well. In this section, we propose to abstract the relationship between the various flows (*energy-energy*, *signal-energy*, and *signal-signal*), either in a single sub-system, or across various sub-systems. This will be achieved by using the Conditional Generative Adversarial

Model (CGAN). We propose to model the system behavior using the relationship between the various flows as shown in Figure 2. Based on the particular system design of CPPS,

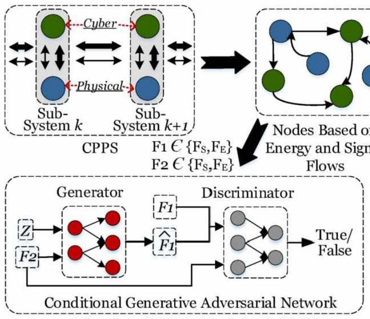


Fig. 2. *CGAN Model of CPPS.*

we can acquire all the signal and energy flows during design time. One can then construct a graph where the various nodes represent the cyber and physical domain components, and the edges represent the signal and energy flow between them (see Figure 3).

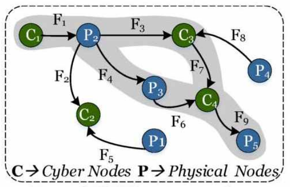


Fig. 3. *Decomposition of CPPS in terms of components and flows.*

The graph created in Figure 3 is then used to list all the flows $F = \{F_1, F_2, \dots, F_n\}$ with $F \in \{F_S, F_E\}$ that are available in the CPPS system. From this, we extract the flow pairs (F_i, F_j). Each pair is then supplied to the CGAN to model $Pr(F_i|F_j = f_j)$ or $Pr(F_j|F_i = f_i)$. We can infer that a high value of conditional probability indicates a strong relationship between two flows (given knowledge of one of the flows). Based on these distributions, we can therefore analyze the relationship between various flows from a security perspective. For example, using $Pr(F_1|F_9)$ (see Figure 3), one can answer questions similar to the following: Is data in F_1 (cyber domain) being leaked from F_9 (physical domain)? Can F_9 be used to monitor any attacks in the integrity of the flow path from node C_1 to P_5? Various other metrics may also be created using the conditional probability values (e.g., mutual information metrics of side channel attacks).

In summary, the proposed CGAN-based security model provides a theoretical foundation to enable a system-level methodology for the design and analysis of CPPS against cross-domain attacks. To the best of our knowledge, this is the first effort towards building a security analysis methodology for CPPS based on CGAN.

III. CGAN Model Generation

To address the need for a design-time security analysis tool for CPPS, we propose to use a two-step method which generates the proposed CGAN-based security model from a given CPPS model, as shown in Figure 4. The two steps include 1) Graph Generation and 2) CGAN-based Security Model Generation. In order to develop the graph from the existing CPPS, the graph generation algorithm takes the design time CPPS architecture as input with information about all the sub-systems $\{Sub_1, Sub_2, \dots, Sub_n\}$, cyber C and physical P domain components with each sub-system, and the corresponding energy flow F_E and signal flow F_S data among each sub-system.

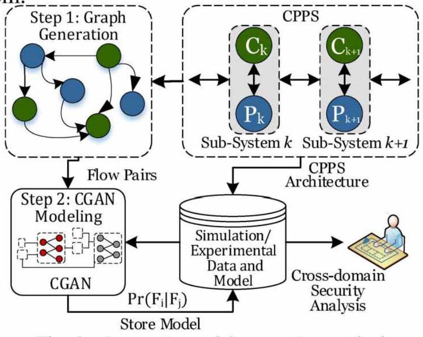


Fig. 4. *Automatic model generation method.*

Step 1: *CPPS graph generation based on signal and energy flows.* First, it is necessary to have a list of all the signal and energy flows between cyber and physical components and also between sub-systems.

Algorithm 1: CPPS graph and flow pairs generation.

```
Input: CPPS Architecture Data: $Sub, C, FP, F_S, F_E$
Input: Historical Data: $Data$
Output: Flow Pair List: $FP_T$
1  Initialize $G_{CPPS}$ with $V$ nodes and its adjacency list, $E$
2  Initialize $FP_F$ and $FP_T$ as flow pairs $(F_i, F_j)$ list
3  Remove feedback loops to make signal/energy flows directed
4  foreach Subsystem $S \in Sub$ do
5      Node list $Q$=(Add all $P_i \in S$ and $C_i \in S$)
6  foreach Node $v \in Q$ do
7      Add $v$ in $G_{CPPS}$
8      foreach Node $u \in Q \setminus v$ do
9          if $F_S$ or $F_E$ exists between $v$ and $u$ then
10             Add $u$ into adjacency list of $v$, $E[v]$
11 foreach Flow $F_{1_i} \in E$ do
12     foreach Flow $F_{2_j} \in (E \setminus F_{1_i})$ do
13         if head of $F_{2_j}$ is reachable from tail of $F_{1_i}$ using DFS
           then
14             $FP_{i,j} = (F_1, F_2)$, $FP_F = FP_F \cup FP_{i,j}$
15 foreach $FP_{i,j} = (F_{1_i}, F_{2_j}) \in FP_F$ do
16     if $FP_{i,j} \in Data$ then
17         $FP_T = FP_T \cup (FP_{i,j})$
18 return $FP_T$
```

The design-time CPPS architecture specifications will consist of this information and we will convert it into a graph-based model. A graph would allow us to select the energy and signal flow pairs necessary to create the CGAN model in the next step. We will denote the corresponding graph generated in this step as G_{CPPS} consisting of nodes made of cyber and physical components. Building the graph, G_{CPPS}, given the design-time architecture of the CPPS, is straightforward and presented in Algorithm 1. After building G_{CPPS}, in Lines 11 to 17 we reduce the graph to find only relevant pairs that can be modeled using CGAN. **Step 2:** *CGAN model generation.* We assume that the given G_{CPPS} is a black box with only input/output flows. There may exist historical data of the CPPS that we can directly use (e.g., testing/runtime data of a system). The $Pr(F_1|F_2)$ learned by the CGAN relies on the training data. In CPPS, this training data is bound by either the architecture or the production capability. Hence, during design-time security analysis, a large amount of training data is required. This may be a drawback of the CGAN-based modeling, but this problem persists in other data collection methods as well. In CPPS, there are some limitations to the access of the high-level data which needs to be protected, such as product specification, due to the mechanical structure of the sub-systems. This allows us to gather training data within the specified bound, and estimate $Pr(F_1|F_2)$ using our model. The amount of data given for training can also be modified according to the attacker capability or attack detection model's resources, such as sub-system type, money or time. Algorithm 2 is used to train the CGAN. The algorithm takes the flow pair list FP_T generated in Algorithm 1 and the data corresponding to the flows in the CPPS.

```
Algorithm 2: CGAN Model Generation and Storage
   Input: Flow Pairs: $FP_T$, Historical Data: $Data$
   Input: CGAN Training Parameters: Batch Size $n$, Step Size $k$,
          Number of Iterations $Iter$
   Output: Generator, Discriminator: $G$, $D$
1  foreach $FP_j \in FP_T$ do
2      Extract flows corresponding to $(F_1, F_2)$ of $FP_j \in Data$
3      for $Iter\ steps$ do
4          for $k\ steps$ do
5              Acquire $n$ mini-batch noise samples
                 $\{z_1, z_2, \dots, z_n\}$ from $Pr(Z)$
6              Acquire $n$ mini-batch noise samples
                 $\{f_{1_1}, f_{1_2}, \dots, f_{1_n}\}$ from $Pr_{data}(F_1)$
7              Corresponding to $\{f_{1_1}, f_{1_2}, \dots, f_{1_n}\}$, acquire
                 $\{f_{2_1}, f_{2_2}, \dots, f_{2_n}\}$ from $Pr_{data}(F_2)$
8              Update $D$, by ascending its stochastic gradient:
                 $\nabla_{\theta_d} \frac{1}{n} \sum_{i=1}^{n} [logD(f_{1_i}|f_{2_i}) + log(1 - D(G(z_i|f_{2_i}))]$
9          Acquire $n$ mini-batch noise samples $\{z_1, z_2, \dots, z_n\}$
             from $Pr(Z)$ and the $\{f_{2_1}, f_{2_2}, \dots, f_{2_n}\}$ used in the
             line 17
10         Update $G$, by descending its stochastic gradient:
             $\nabla_{\theta_g} \frac{1}{n} \sum_{i=1}^{n} [log(1 - D(G(z_i|f_{2_i}))]$
11 return $G$ that estimates $Pr(F_i|F_j)$, and $D$
```

For each of the flow pairs available in FP_T, Lines 1 to 10, which are based on [11], [12], iteratively take sample flows from F_i and F_j, and train D by stochastic gradient ascent (Line 8), and train G by stochastic gradient descent (Line 10). The number of steps and the iterations to be performed depends on the assumptions about the attacker and can be easily modified accordingly. At the end, G learned for each flow pair is returned and stored. This model effectively will learn the conditional probability $Pr(F_i|F_j)$ [11], [12].

The convergence of G to the historical data has been extensively proven in [11], [12]. Then based on the assumption of how much actual data an attacker can acquire, or how much data an attack detection model can acquire, $Pr(F_i|F_j)$ can be estimated accordingly by adjusting D in the training phase.

IV. Case Study and Analysis

As a case study, we provide the CGAN-based modeling for security analysis of a sub-system of a CPPS. The CPPS sub-system selected is a fused modeling deposition-based additive manufacturing system, also known as a 3D printer. Additive manufacturing has been predicted to be one of the enabling technologies for the next generation of CPPS due to its rapid prototyping and distributed manufacturing capability. However, many emerging threats to this CPPS subsystem has also been highlighted [13], [14]. Hence, we will demonstrate how some security analysis may be performed in this sub-system using the proposed CGAN-based modeling.

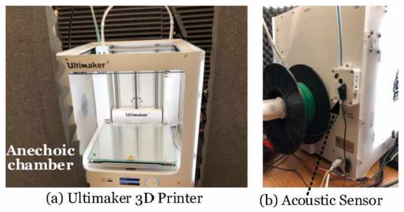


Fig. 5. Additive manufacturing as a sub-system for security analysis.

The state-of-the-art cartesian 3D printer model consists of four stepper motors as shown in Figure 5. Three of them are used to provide printer nozzle movement in the X, Y, and Z directions, while the fourth motor is used to extrude the filament while printing. The speed and direction of all the stepper motors are controlled by cyber domain instructions written with G-code, a programming language widely used in industrial systems to tell a tool what to do and how to do it, along with M-code, auxiliary commands for miscellaneous machine functions. Our experimental setup is enclosed in a makeshift anechoic chamber to isolate the noise from the environment and other components in the lab. A contact microphone (C411 L) is attached to the back of the 3D printer to acquire the acoustic energy flow dissipated to the environment.

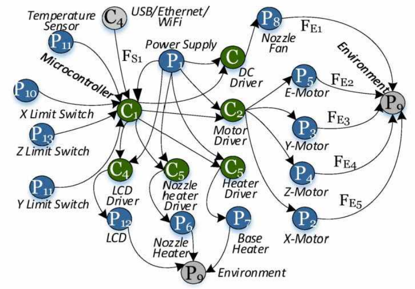


Fig. 6. (G_{CPPS}) Graph generation for additive manufacturing system based on signal and energy flows.

A. G_{CPPS} Generation

Using Algorithm 1, the additive manufacturing CPPS can be decomposed to its corresponding energy and signal flows in the form of a graph, G_{CPPS} (see Figure 6). One may notice that vertices C_4 and P_9 are not components of the 3D printer. C_4 represents the external signal flows from other sub-systems into the 3D printer. The node P_9, on the other hand, represents the physical environment. Various energy flows that are either intentional or unintentional passing to the environment are encompassed by the edges going towards the node P_9. From this graph, flow pairs P_T is extracted using Algorithm 1.

B. Experimental Data Collection

In our experiment, we have only selected cross-domain flow pairs for security analysis. Specifically, we analyze the acoustic/vibration energy in the physical domain and the G/M-code instructions passed to the 3D printer subsystem from an external node. The G/M-code instructions are the signal flows entering the sub-system from node C_4. In our analysis, we have monitored the energy flows between nodes P_2, P_3, P_4, P_5, P_8 and the node P_9. For ease of training the CGAN, we convert the time-domain acoustic energy flows values into frequency domain values using continuous-wavelet transforms, which preserves the high-frequency resolution in time-domain as well. We obtain a non-uniformly distributed 100 bins $Freq = [freq_1, freq_2, \dots, freq_{100}]$ between 50 and 5000 Hz (this range may be changed for further security analysis purposes). In this case-study analysis, for simplicity, we extract G/M-codes from 3D objects that only move one stepper motor at a time. The G/M code is one-hot encoded based on presence of instructions that run stepper motors X ($[1, 0, 0]$), Y($[0, 1, 0]$) and Z($[0, 0, 1]$), respectively. This encoding is done based on G/M-codes G_t and G_{t-1}. For example, if G_{t-1} is [G_1 $F1200$ $X5$ $Y5$ $Z5$] and G_t is [G_1 $F1200$ $X10$ $Y5$ $Z5$] then encoding for G_t will be $Cond = [1, 0, 0]$ as only stepper motor X will run. For a more thorough security analysis, the one-hot encoding can be extended to consider the combination of signal and energy flows. For example, for three physical components and their combination, the one-hot encoding can be of size $2^3 = 8$.

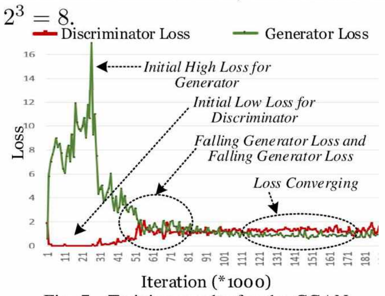


Fig. 7. Training results for the CGAN.

C. CGAN Modeling

Using the one-hot encoding obtained in Section IV-B as conditions $Cond$, we have trained CGAN to estimate $Pr(Freq|Cond)$. In doing so, we can derive a trained conditional density function to find the probability of a particular value of frequency component f_{req} given which stepper motor is running. These conditions are generated from the signal flows coming from node C_4, and frequencies extracted from the acoustic energy flows going to the environment node P_9 from the various nodes. Hence, the conditional density function estimates the relation between the signal flow from node C_4 to C_1 and energy flows from nodes P_2, P_3, P_4, P_5, P_8 to the node P_9.

Figure 7 shows the training results for the CGAN. On the *X-axis*, the iteration number is increasing. With the increasing iteration, however, the more signal and energy pair data are also incorporated. We can observe that initially, G's loss is high, whereas D's loss is low. However, over more iterations and data, the G's loss decreases, making it difficult for D to know whether the data generated is real or fake, and hence increasing the loss of D.

D. Security Analysis Results

In our experiment we will demonstrate how the CGAN modeling may be used for security analysis concerned with *confidentiality* breach through the side-channels, and design of *integrity* and *availability* attack detection on the physical components of the 3D printer sub-system using the physical domain (or the same side-channels).

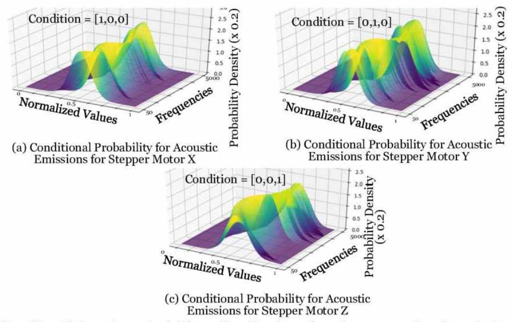
Algorithm 3: Security analysis methodology

```
Input: $D$,$G$,$Z$,$Cond$, $GSize$, $X_{test} \in \mathbb{R}^{\mathrm{LxM}}$
Input: Frequency Feature Indices: $FtIndices \in \mathbb{R}^{\mathrm{K}}$
Input: Parzen Window Width: $h$
Output: Likelihood Metrics: AvgCorLike, AvgIncLike
1  Initialize AvgCorLike and AvgIncLike as $\mathbb{R}^{\mathrm{NxK}}$ matrix
2  foreach $C_i \in Cond$ do
3      $CorLike = 0$, $CorNum = 0$
4      $IncLike = 0$, $IncNum = 0$
5      foreach $FtIdx \in FtIndices$ do
6          $X_G$ = generated $GSize$ samples from $G(Z|C_i)$
7          foreach $X_{test}^{l,FtIdx}$, where $l \in [1, \dots, L]$ do
8              FtDistr = ParzenDensity($X_G^{FtIdx}$, 'Gaussian', $h$)
9              LogLike = FtDistr.score($X_{test}^{l,FtIdx}$)
10             Like = $exp(LogLike) * h$
11             if $Label(X_{test}^{l,FtIdx}) == Condi$ then
12                 CorLike += Like, CorNum += 1
13             else
14                 IncLike += Like, IncNum += 1
15         CurrAvgCorLike=CurrAvgCorLike ∪ {CorLike / CorNum}
16         CurrAvgIncLike=CurrAvgIncLike ∪ {IncLike / IncNum}
17     AvgCorLike[i] = CurrAvgCorLike
18     AvgIncLike[i] = CurrAvgIncLike
19     Reset CurrAvgCorLike and CurrAvgIncLike
20 return AvgCorLike, AvgIncLike
```

Both of this analysis have one thing common, the conditional relationship between the energy flows given the signal flows. In this section, we will show how the CGAN model (Discriminator D, Generator G, and Noise Z) can be used to check this conditional relation for security analysis. The preliminary algorithm used for security analysis shown in Algorithm 3 (this may be changed for more complex signal flow analysis but can still use the same CGAN). Algorithm 3 computes security metrics based on the average likelihood of test samples. If a test sample x has a high likelihood for a specific conditional distribution $Pr(x|Cond_i)$ and its actual label is $Cond_i$ then it means that there is a strong relationship between $Cond_i$ and the x. This type of metric can be used to evaluate the system's confidentiality vulnerability level or perhaps to detect an integrity or availability attack.

For each condition label $Cond_i$, we first generate samples X_G with $G(Z|Cond_i)$. Then for a given Parzen window size h, and current feature index $FtIdx$, we create an estimated conditional distribution $FtDistr = Pr(X_G^{FtIdx}|Cond_i)$ via the Parzen Gaussian Window method (Line 8). For each frequency feature, we derive the corresponding test samples from our test batch X_{test} (Line 7), and for each test sample, we update two metrics based on the likelihood.

The first metric $CorLike$ is the total likeliness that this test sample belongs to $FtDistr$ and the other $IncorLike$ refers to the total likeliness that it does not belong to $FtDistr$. We then average the two metrics according to the number of test samples per feature (Lines 15-16).



(a) Conditional Probability for Acoustic Emissions for Stepper Motor X

(b) Conditional Probability for Acoustic Emissions for Stepper Motor Y

(c) Conditional Probability for Acoustic Emissions for Stepper Motor Z

Fig. 8. Conditional probability distribution for the acoustic signal (h=0.2).

In the outermost loop, based on the conditions, we update two sets $AvgCorLikes$ and $AvgIncLikes$, with the corresponding sets of averaged metric values created in the inner loops. Higher values in the first set mean that our model has learned a better relationship between data (e.g., features) and their correct conditions (e.g., running motors). However, higher values in the second set mean that our model has learned unexpected and potentially unrealistic relationships between data and other conditions.

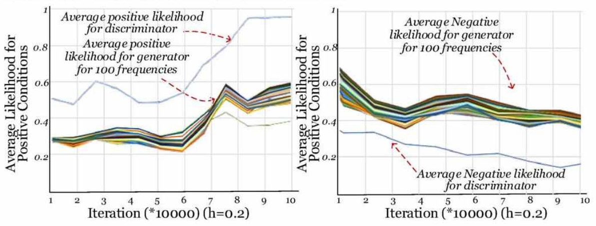


Fig. 9. Average correct and incorrect likelihood values for $Cond$=$[1, 0, 0]$.

The conditional density functions estimated by the CGAN is shown in Figure 8. The frequency magnitudes for the $Freq = [freq_1, freq_2, \ldots, freq_{100}]$ are scaled between 0 and 1. The parzen window width h value used for the Gaussian kernel density estimation is 0.2. Hence, the actual probability of the frequency values is obtained by multiplying it by 0.2. In Figure 9, the average correct and incorrect likelihood values are presented for the condition ($[1, 0, 0]$). As it can be seen, over increasing iterations, the positive likelihood averages improve. This shows that the generator is able to accurately learn the conditional distribution of the acoustic emissions according to the signal flows.

Based on the density function estimated using the generator of the CGAN, we have used the security analysis algorithm 3 to calculated the average correct and incorrect likelihoods of the emissions given the three conditions (presence of X, Y or Z motor movement in the G/M-code). Using this, a CPPS designer can estimate if an attacker is able to estimate the G/M-code based on the acoustic emissions. For example, in table I, we have presented the average correct and incorrect likelihoods of a single feature in the frequency domain based on the G/M-code related to the X, Y, or stepper motor. The table shows that an attacker can estimate condition 3, which is the presence of the Z-motor movement in the G/M-code, better than the other conditions (presence of X or Y motor movement). Moreover, if a designer needs to create an integrity

TABLE I
AVERAGE CORRECT (COR) AND INCORRECT (INC) LIKELIHOOD OF ACOUSTIC ENERGY FLOWS GIVEN VARIOUS CONDITIONS FOR A SINGLE FEATURE (H=PARZEN WINDOW).

	h=0.2		h=0.4		h=0.6		h=0.8		h=1	
	Cor	Inc	Cor	Inc	Cor	Inc	Cor	Inc	Cor	Inc
Cond1	0.6000	0.2245	0.6000	0.3247	0.6069	0.3634	0.6293	0.3783	0.6437	0.3856
Cond2	0.5750	0.3887	0.5750	0.3961	0.5750	0.3974	0.5750	0.3982	0.5532	0.3978
Cond3	0.6556	0.3876	0.6556	0.3956	0.6556	0.3979	0.6601	0.3983	0.6556	0.3985

and availability attack detection model to detect attacks on individual components (X, Y or Z motor) using the side-channels, he/she will be able to estimate the performance of such a model using the CGAN model.

V. Conclusion

We have presented GAN-Sec, a Conditional Generative Adversarial Network (CGAN) based modeling approach for security analysis of Cyber-Physical Production Systems (CPPS). To do this, GAN-Sec abstracts the system in terms of its flows and estimates the conditional distribution between them using a CGAN model. We have used GAN-Sec to analyze the security of an additive manufacturing CPPS sub-system. The promising results indicate that GAN-Sec is applicable in analyzing the cross-domain security of CPPS.

VI. Acknowledgment

This work was partially supported by **NSF** CPS CNS-1546993 and CMMI-1739503 grants.

References

[1] L. Monostori, "Cyber-physical production systems: Roots, expectations and r&d challenges," *Procedia CIRP*, 2014.

[2] S. R. Chhetri *et al.*, "Kcad: kinetic cyber-attack detection method for cyber-physical additive manufacturing systems," in *Proceedings of the 35th International Conference on Computer-Aided Design*, ACM, 2016.

[3] S. D. Applegate, "The Dawn of Kinetic Cyber," in *2013 5th International Conference on Cyber Conflict (CyCon)*, IEEE, 2013.

[4] S. R. Chhetri *et al.*, "Confidentiality breach through acoustic side-channel in cyber-physical additive manufacturing systems," *ACM Transactions on Cyber-Physical Systems*, 2018.

[5] S. R. Chhetri, S. Faezi, and M. A. Al Faruque, "Information leakage-aware computer-aided cyber-physical manufacturing," *IEEE Transactions on Information Forensics and Security*, 2018.

[6] D. Kushner, "The real story of stuxnet," *Spectrum, IEEE*, 2013.

[7] R. M. Lee, M. J. Assante, and T. Conway, "German Steel Mill Cyber Attack," *Industrial Control Systems*, vol. 30, 2014.

[8] A. Cardenas, S. Amin, *et al.*, "Challenges for securing cyber physical systems," in *Workshop on future directions in cyber-physical systems security*, 2009.

[9] J. Sztipanovits, T. Bapty, S. Neema, L. Howard, and E. Jackson, "OpenMETA: A model-and component-based design tool chain for cyber-physical systems," in *Joint European Conferences on Theory and Practice of Software*, pp. 235–248, Springer, 2014.

[10] R. Akella, H. Tang, *et al.*, "Analysis of information flow security in cyber–physical systems," *International Journal of Critical Infrastructure Protection*, 2010.

[11] I. Goodfellow, J. Pouget-Abadie, *et al.*, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014.

[12] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[13] L. Sturm, C. Williams, *et al.*, "Cyber-physical vunerabilities in additive manufacturing systems," *Context*, 2014.

[14] S. E. Zeltmann, N. Gupta, *et al.*, "Manufacturing and Security Challenges in 3D Printing," *JOM*, 2016.