A Practical Side-Channel Based Intrusion Detection System for Additive Manufacturing Systems

Sizhuang Liang
Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, USA
liangsizhuang@gatech.edu

Xirui Peng
Mechanical Engineering
Georgia Institute of Technology
Atlanta, USA
aaronpeng@gatech.edu

H. Jerry Qi
Mechanical Engineering
Georgia Institute of Technology
Atlanta, USA
qih@me.gatech.edu

Saman Zonouz
Electrical and Computer Engineering
Rutgers University
Piscataway, USA
saman.zonouz@rutgers.edu

Raheem Beyah
Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, USA
rbeyah@ece.gatech.edu

***Abstract*—We propose NSYNC, a practical framework to compare side-channel signals for real-time intrusion detection in Additive Manufacturing (AM) systems. The motivation to develop NSYNC is that we find AM systems are asynchronous in nature and there is random variation in timing in a printing process. Although this random variation, referred to as time noise, is very small compared with the duration of a printing process, it can cause existing Intrusion Detection Systems (IDSs) to fail.**

To deal with this problem, NSYNC incorporates a dynamic synchronizer to find the timing relationship between two signals. This timing relationship, referred to as the horizontal displacement, can not only be used to mitigate the adverse effect of time noise on calculating the (vertical) distance between signals, but also be used as indicators for intrusion detection. An existing dynamic synchronizer is Dynamic Time Warping (DTW). However, we found in experiments that DTW not only consumes an excessive amount of computational resources but also has limited accuracy for processing side-channel signals. To solve this problem, we propose a novel dynamic synchronizer, called Dynamic Window Matching (DWM), to replace DTW.

To compare NSYNC against existing IDSs, we built a data acquisition system that is capable of collecting six different types of side-channel signals and performed a total of 302 benign printing processes and a total of 200 malicious printing processes with two printers. Our experiment results show that existing IDSs leveraging side-channel signals in AM systems can only achieve an accuracy from 0.50 to 0.88, whereas our proposed NSYNC can reach an accuracy of 0.99.

***Index Terms*—side channel, intrusion detection, dynamic synchronization, additive manufacturing, cyber-physical system**

I. Introduction

Side channels are gaining popularity for intrusion detection in Additive Manufacturing (AM) systems due to their non-invasiveness and often air-gapped threat models [4], [5], [9], [12], [13], [18], [27]. To perform intrusion detection in AM systems, many existing Intrusion Detection Systems (IDSs) compare an observed side-channel signal against a reference signal point by point or window by window. For each pair of points or windows, a distance is calculated between the signals to determine if they are similar or not [4], [5], [9], [12], [13], [18] (the acoustic layer in [4]), or a classifier can be employed to determine whether the signals are alike or not [4] (the spatial layer in [4]).

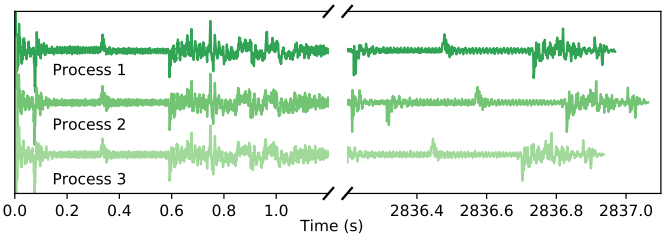


Fig. 1: Side-channel signals for three printing processes using the same G-code file and the same printer. The signals are aligned at the beginning. The misalignment in the end is caused by time noise.

This approach to compare signals works well if the two signals are aligned for every pair of points or windows. It appears that when the same printing process is performed on the same AM system, the timing should be the same. If this assumption is true, when an observed signal is aligned with a reference signal at the beginning, they should align at other points. However, our experiments show that this assumption is not true. Fig. 1 shows side-channel signals from three printing processes with the same G-code file and the same printer. Although the signals are aligned at the beginning, the signals do not end at the same time.

AM systems are asynchronous. When executed multiple times, the duration for the same instruction can vary slightly. In addition, there can be random gaps between instructions. This random variation in timing is referred to as *time noise*. Time noise can be a result of frame drops in data acquisition systems, mechanical and thermal delays in devices, and task scheduling in operating systems (if equipped).

Time noise can invalidate any IDS that is based on comparing a side-channel signal against a reference signal point

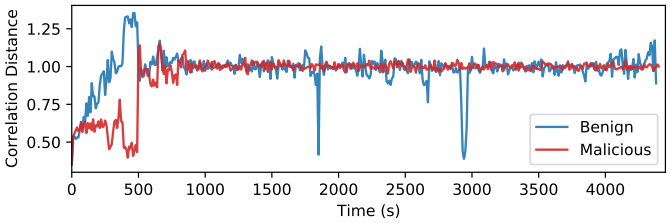


Fig. 2: Correlation distances of a benign process and a malicious process. Due to time noise, the distances of a benign process are very large, and even larger than the distances of a malicious process.

by point or window by window. When comparing two signals that are out of alignment, the distances can become very large. Fig. 2 shows the correlation distances of a benign process and a malicious process. Without the consideration of time noise, we can see that the distances of the benign process are as large as the distances of the malicious process.

A practical IDS using side channels must be able to tolerate time noise. One approach to tolerate time noise is to dynamically synchronize (or align) two signals, and an existing method to do so is Dynamic Time Warping (DTW) [20], [22]. However, DTW does not natively support real-time operations, consumes excessively computational resources, and has limited accuracy for side-channel signals in AM systems, as will be demonstrated in Section VIII-E.

To overcome the problems of DTW, we propose a novel algorithm called Dynamic Window Matching (DWM) to replace DTW. DWM finds the timing relationship between two signals by establishing a sliding window for each signal. As the pair of windows slides across the signals, the relative displacements (aka horizontal displacements) between the windows are determined by Time Delay Estimation (TDE). To stabilize the process, we introduce bias in the TDE process, and we introduce inertial when adjusting the relative displacements between the windows. DWM is a window-by-window algorithm. In contrast, DTW is a point-by-point algorithm.

For a complete IDS, we propose NSYNC (Noise SYNC), a framework to practically compare a side-channel signal against a reference signal for real-time intrusion detection in AM systems. NSYNC starts with two signals that are aligned at the beginning. The horizontal displacements between the two signals are then determined by either DTW or DWM. A comparator then generates the vertical distances for each pair of points or windows. Finally, a discriminator looks at both the horizontal displacements and the vertical distances to automatically determine if there is an intrusion.

In NSYNC, the thresholds are learned by One-Class Classification (OCC) [19]. In contrast, many existing IDSs either use binary classification [4], [9], [27] (the spatial layer in [4]) or magic numbers for thresholds [4], [5] (the acoustic layer in [4]). Some existing IDSs do not have an automatic decision module [12]. Binary classification requires knowing the malicious processes in advance, which can be impractical to achieve. In contrast, OCC does not require such knowledge.

To our best knowledge, we are the first one in the literature to consider time noise for side-channel based intrusion detection for AM systems. Our contributions are as follows:

- We propose a novel algorithm, called Dynamic Window Matching (DWM), to dynamically align two signals in real time.
- We present NSYNC, a practical framework to compare side-channel signals for intrusion detection in AM systems.
- We empirically demonstrate that signals to be compared lose synchronization over time due to time noise, and, if not compensated for, will cause inaccurate classification.

II. Background Information

In this section, we introduce background information about AM and side channels.

A. Additive Manufacturing

Additive Manufacturing (AM), also known as 3D printing, refers to a manufacturing process where material is joined together layer by layer to make objects directly from design models [3]. The operation of an AM system is also called printing and the machine where materials are joined together is also called a printer. There are many types of AM systems. This paper focuses on the most common type of AM systems, namely Fused Deposition Modeling (FDM) [10].

AM systems are controlled by computers and typically require programming to work. G-code is a common programming language supported by almost all FDM systems. G-code instructions specify the target coordinates and target velocities of all movements in a printing process. However, G-code instructions do not specify timing. An AM system has freedom in determining the acceleration for any given G-code instruction. As a result, it could take a slightly different amount of time for the same instruction to be executed. In addition, when a G-code instruction is sent to an AM system for execution, the instruction is put into a queue. The AM system is allowed to delay the execution of any instruction. As a result, there can be random gaps between G-code instructions, although this gap is typically very small.

B. Side Channels

Side channels are unintentional means of communication by which information about a computer or a cyber-physical system can be leaked to an outsider [8][1]. Side-channel signals are the carriers of information inside channels. AM systems have a variety of side channels. For example, when an AM system is printing an object, the system emits acoustic waves [1], [15], [24] and electromagnetic waves (including quasi-static electric fields and quasi-static magnetic fields) that can be sensed to infer information about the printing process. The acceleration of any moving part in the AM system can be measured by accelerometers to infer information about the printing

[1]In this paper, side channels exclusively refer to analog side channels. In addition to analog side channels, there are other types of side channels, such as cache in a shared memory system. In this paper, we are only concerned with analog side channels.

process [4], [12]. Other examples of side channels in an AM system include, but are not limited to, power consumption measured by power sensors [13], [18], temperature measured by thermometers, optical videos captured by cameras [12], [27], and infrared videos captured by infrared cameras [11].

III. Related Work

This section describes existing IDSs that use side channels to perform intrusion detection in AM systems. We do not find any existing IDS that is aware of time noise.

Chhetri *et al.* came up with the idea of using the acoustic side channel in a printing process to detect zero-day cyberattacks on AM systems [9]. They first use machine learning to estimate the velocities and positions of the nozzle by analyzing the acoustic side-channel signal. Afterwards, they interpret the G-code instructions to determine the intended velocities and positions. Finally, the estimated velocities and positions are compared against the intended ones to determine if there is an intrusion. However, this method is not practical unless the side-channel signal can be segmented with each segment corresponding to a single G-code instruction. Currently, it is not clear how to perform this segmentation.

Bayens *et al.* presented a method to compare an acoustic side-channel signal against a reference signal window by window with Dejavu, a music retrieval engine that is similar to Shazam [26], to detect malicious infill patterns [4]. They also presented a method to compare the position signal (the position of the nozzle with respect to time) against a pre-recorded reference signal layer by layer to detect intrusion. However, they used binary classification in this process, which requires knowing the malicious processes in advance.

Moore *et al.* proposed an IDS that observes electric currents delivered to actuators, and compares the observed signal against a pre-recorded reference signal point by point to detect malicious activities [18]. As with other existing IDSs, this IDS is not aware of time noise. In addition, the current sensors are invasive. In fact, it is very hard to access the wire of motors in many commercially available printers.

Later, Gatlin *et al.* improved Moore's IDS [18] in two ways [13]. On the one hand, the new IDS analyzes the electric currents in the Z motor to determine the moments when a layer change happens. On the other hand, instead of comparing power side-channel signals directly, the new IDS first extracts fingerprints of the power side-channel signals for each layer and then compares the fingerprints. An intrusion is declared if the layer changing moments differ from the expected values by pre-determined thresholds, or the number of fingerprint mismatches exceeds pre-determined thresholds.

Gao *et al.* presented a process monitoring system that observes multiple side-channel signals to safeguard AM systems against cyberattacks [12]. Based on the observed signals, state variables, such as the position and velocity of the nozzle, the height of each layer, and the fan speed, are estimated. Afterwards, the G-code file is interpreted to obtain the intended state variables. The estimated state variables are compared against the intended ones layer by layer, and the comparison results are displayed. There is no module in the IDS to automatically determine if a printing process is benign or malicious. To segment side-channel signals layer by layer, they used a dedicated accelerometer on the printing bed to determine the moments when a layer change happens.

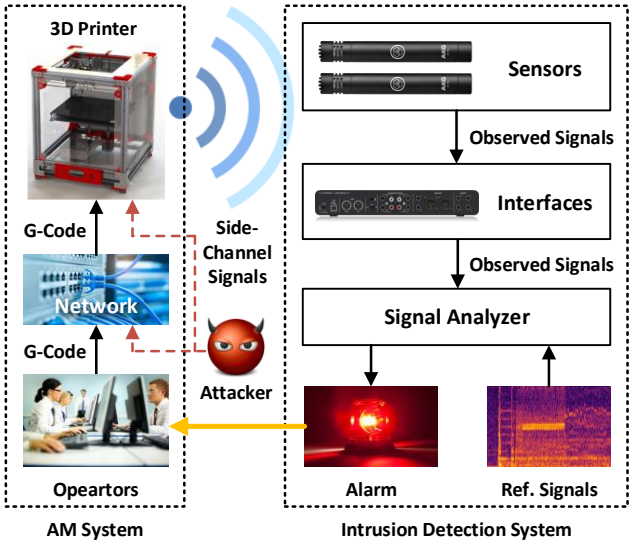


Fig. 3: Threat model. An attacker attempts to maliciously modify the G-code file in the network or the firmware in the printer. An air-gapped intrusion detection system monitors the side-channel signals in the printing process to determine if there is any attack.

Belikovetsky *et al.* presented an IDS based on the acoustic side channel and the Principle Component Analysis (PCA) [5]. In this IDS, an acoustic side-channel signal is first transformed into a spectrogram, which is then compressed by PCA into a signal with only three channels [5]. The compressed signal is compared against a reference signal (derived from a benign printing process) to detect intrusion.

IV. Threat Model

The threat model, as shown in Fig. 3, is similar to the threat models found in [4], [5], [9], [12], [13], [18], [27]. In the threat model, an AM system is manufacturing a functional object. An attacker wants to compromise the structural integrity of the printed object without being detected. We assume that the attacker can either modify the G-code instructions to be sent to the printer or the firmware of the printer. By modifying the firmware, the printer behaves maliciously despite being sent benign G-code instructions. The attacker knows how to strategically modify the G-code or the firmware to weaken the structural integrity of the object and let the object pass existing quality checks, such as the attack demonstrated in [6].

In the AM system, an air-gapped IDS using side channels is deployed. The IDS is composed of an analyzer, reference signals, and sensors to observe side-channel signals. We assume that the attacker cannot tamper with any element in the IDS. The analyzer, essentially a digital computer, continuously compares the observed signals against the reference signals. If an observed signal is determined to be different from

its corresponding reference signal at any moment, then an intrusion is declared and the IDS alerts AM operators, and automatically stops the printing process if necessary.

Acquirement of Reference Signals. A reference signal is a side-channel signal recorded from a benign process. A challenge to obtain a reference signal is to ensure that it indeed comes from a benign process. One way to do this is to subject the printed object to stringent tests and if the printed object passes the tests, the printing process can be considered benign [5], [13]. An alternative way to obtain a reference signal is to simulate a process with its G-code file [9], [12].

Requirements of Side Channels. In order for the aforementioned IDS to work, the selected side-channel signals must be highly correlated with the state of the printer. On the one hand, when the state of the printer is modified, this change should be reflected in the side-channel signals. Otherwise, the IDS will likely result in false negatives. On the other hand, when the state of the printer is not altered, the side-channel signals should remain almost the same. Otherwise, the IDS will likely result in false positives.

V. Signal Processing for NSYNC

In this section, we introduce signal notation and Time Delay Estimation (TDE). They are needed to understand NSYNC.

A. Signal Notation

A signal is denoted by $\boldsymbol{x}[n]$, where $n = 0, 1, \cdots, N-1$ is the time index and N is the number of samples. Suppose f_s is the sampling frequency. Then, n corresponds to time $t = n/f_s$. The whole signal $\boldsymbol{x}[n], n = 0, 1, \cdots, N-1$ can be simply denoted by $\boldsymbol{x}$.

For each time index n, $\boldsymbol{x}[n]$ is a vector of one or more components. For example, the acceleration measured by an accelerometer is a vector of three components (a_x, a_y, a_z). Each component of $\boldsymbol{x}[n]$ as a function of n is defined as a channel. The number of channels in $\boldsymbol{x}$ is denoted by C.

We use $\boldsymbol{x}[n, c]$ to refer to the nth sample at the cth channel, where $c = 0, 1, \cdots, C-1$. We use $\boldsymbol{x}[n_1 : n_2]$ to refer to a slice of $\boldsymbol{x}$ from index n_1 (inclusive) to index n_2 (exclusive). We use $\boldsymbol{x}[:, c]$ to refer to all samples at the cth channel.

B. Time Delay Estimation (TDE)

Suppose $\boldsymbol{x}$ and $\boldsymbol{y}$ are signals with finite samples and the length of $\boldsymbol{x}$ is longer than that of $\boldsymbol{y}$. *Time Delay Estimation* (TDE) is a process to determine the best location of $\boldsymbol{y}$ in $\boldsymbol{x}$, assuming that $\boldsymbol{y}$ appears in $\boldsymbol{x}$ once. TDE is the prerequisite of DWM, which will be introduced in the next section. One way to perform TDE is the sliding method [7], [16].

The Sliding Method. Suppose the length of $\boldsymbol{x}$ is N_x, the length of $\boldsymbol{y}$ is N_y ($N_x \geq N_y$), and the number of channels for both $\boldsymbol{x}$ and $\boldsymbol{y}$ is C. One way to perform TDE is to compare $\boldsymbol{y}$ against $\boldsymbol{x}[n : n + N_y]$ for $n = 0, 1, \cdots, N_x - N_y$. For each n, we measure the similarity between $\boldsymbol{x}[n : n + N_y]$ and $\boldsymbol{y}$ by a score $\boldsymbol{s}[n]$. The similarity scores $\boldsymbol{s}[n]$ form a new array with a length of $N_x - N_y + 1$. We have

$$\boldsymbol{s}[n] = f(\boldsymbol{x}[n : n + N_y], \boldsymbol{y}), n = 0, 1, \cdots, N_x - N_y, \quad (1)$$

where f is a function to calculate the similarity score, also known as the similarity function.

Since a higher similarity score indicates more similarity, the best location of $\boldsymbol{y}$ in $\boldsymbol{x}$ is given by

$$n_{\text{delay}} = \underset{n}{\operatorname{argmax}}\, \boldsymbol{s}[n], \quad (2)$$

which means that $\boldsymbol{y}[0]$ corresponds to $\boldsymbol{x}[n_{\text{delay}}]$.

Similarity Functions. Suppose $\boldsymbol{u}$ and $\boldsymbol{v}$ are two 1-D vectors of the same length N. One similarity function is the correlation coefficient [17]

$$f(\boldsymbol{u}, \boldsymbol{v}) = \frac{(\boldsymbol{u} - \mu_u) \cdot (\boldsymbol{v} - \mu_v)}{\|\boldsymbol{u} - \mu_u\|_2 \cdot \|\boldsymbol{v} - \mu_v\|_2}, \quad (3)$$

where $\|\cdot\|_2$ is the L2 norm operator and

$$\mu_u = \frac{1}{N} \sum_{n=0}^{N-1} \boldsymbol{u}[n], \qquad \mu_v = \frac{1}{N} \sum_{n=0}^{N-1} \boldsymbol{v}[n]. \quad (4)$$

The similarity function in Eq. (3) requires the two inputs be 1-D, whereas $\boldsymbol{x}[n : n + N_y]$ and $\boldsymbol{y}$ are 2-D, unless $C = 1$. When $C > 1$, we calculate the similarity score between $\boldsymbol{x}[n : n + N_y, c]$ and $\boldsymbol{y}[:, c]$ for $c = 0, 1, \cdots, C-1$, and average the similarity scores across the channels. We found in experiments that this approach can reach a relatively high Signal-to-Noise Ratio (SNR) because it discards channel-wise information, and focuses on time-wise information.

In NSYNC, we use the correlation coefficient as the default similarity function.

VI. Dynamic Synchronization for NSYNC

Dynamic Synchronization (DSYNC) refers to any process that continuously identifies corresponding points or windows in two signals ($\boldsymbol{a}$ and $\boldsymbol{b}$). DSYNC is needed when there is time noise in $\boldsymbol{a}$ and $\boldsymbol{b}$. In this section, we first discuss Dynamic Time Warping (DTW), an existing method to perform point-based DSYNC. We then discuss Dynamic Window Matching (DWM), a novel method to perform window-based DSYNC.

A. Dynamic Time Warping (DTW)

Point-Based Comparison. Suppose $\boldsymbol{a}$ and $\boldsymbol{b}$ have been aligned at the very beginning with a reasonable accuracy (not necessarily perfect), and we want to compare $\boldsymbol{a}$ and $\boldsymbol{b}$ point by point. One may calculate the distance between $\boldsymbol{a}[n]$ and $\boldsymbol{b}[n]$ for $n = 0, 1, \cdots$. However, due to time noise, the comparison between $\boldsymbol{a}[n]$ and $\boldsymbol{b}[n]$ is meaningless since they may not be corresponding points.

Overview of DTW. Dynamic Time Warping (DTW) is an existing method to find the corresponding points between $\boldsymbol{a}$ and $\boldsymbol{b}$ [22]. DTW requires a distance metric $d(\cdot, \cdot)$ be provided, and then outputs a list of tuples where a tuple (i, j) specifies that $\boldsymbol{a}[i]$ and $\boldsymbol{b}[j]$ are corresponding points.

FastDTW. Due to the time complexity of DTW, a variation of DTW, called FastDTW, is typically used [23]. FastDTW requires an additional parameter called the radius. Because FastDTW is an approximation to DTW, this parameter controls the trade-off between speed and accuracy. We always use the

smallest radius for the fastest speed because it takes a very long time to analyze side-channel signals by FastDTW. In this paper, we simply use DTW to refer to FastDTW.

Online DTW. DTW requires knowing the whole $\boldsymbol{a}$ and the whole $\boldsymbol{b}$ before they can be analyzed. In other words, DTW does not support real-time analysis. Fortunately, there is an ongoing effort to create a version of DTW that supports real-time analysis [21].

Horizontal Displacement $\boldsymbol{h}_{\text{disp}}$. For a tuple (i, j), we define $j - i$ as the horizontal displacement of $\boldsymbol{b}$ with respect to $\boldsymbol{a}$ at index i. In other words, we have $\boldsymbol{h}_{\text{disp}}[i] = j - i$. If there are multiple tuples with the first index being i, such as $(i, j_1), (i, j_2), \cdots, (i, j_{K_i})$, we define

$$\boldsymbol{h}_{\text{disp}}[i] = \frac{1}{K_i} \sum_{k=1}^{K_i} j_k - i. \tag{5}$$

B. Dynamic Window Matching (DWM)

Window-Based Comparison. Instead of comparing $\boldsymbol{a}$ and $\boldsymbol{b}$ point by point, we can alternatively compare $\boldsymbol{a}$ and $\boldsymbol{b}$ window by window. To be specific, we calculate the distance between $\boldsymbol{a}\{i\}$ and $\boldsymbol{b}\{i\}$, where

$$\boldsymbol{a}\{i\} = \boldsymbol{a}[i \cdot n_{\text{hop}} : i \cdot n_{\text{hop}} + n_{\text{win}}], \tag{6}$$

$$\boldsymbol{b}\{i\} = \boldsymbol{b}[i \cdot n_{\text{hop}} : i \cdot n_{\text{hop}} + n_{\text{win}}], \tag{7}$$

$i = 0, 1, \cdots$ is the window index, n_{win} is the window width, and n_{hop} is the number of samples by which the windows move forward each time. $\boldsymbol{a}\{i\}$ is referred to as the ith window of $\boldsymbol{a}$, whereas $\boldsymbol{b}\{i\}$ is referred to as the ith window of $\boldsymbol{b}$. Due to time noise, the comparison between $\boldsymbol{a}\{i\}$ and $\boldsymbol{b}\{i\}$ can be meaningless, even if $\boldsymbol{a}[0]$ is aligned with $\boldsymbol{b}[0]$ perfectly.

Overview of DWM. To solve this problem, for each $\boldsymbol{a}\{i\}$, instead of comparing it with $\boldsymbol{b}\{i\}$, we attempt to find a better window of $\boldsymbol{b}$ to compare with. Suppose such a window does exist and it can be expressed by

$$\boldsymbol{b}\{i; \boldsymbol{h}_{\text{disp}}[i]\} = \boldsymbol{b}[i \cdot n_{\text{hop}} + \boldsymbol{h}_{\text{disp}}[i] : i \cdot n_{\text{hop}} + \boldsymbol{h}_{\text{disp}}[i] + n_{\text{win}}] \tag{8}$$

where $\boldsymbol{h}_{\text{disp}}[i]$ is referred to as the horizontal displacement of $\boldsymbol{b}$ with respect to $\boldsymbol{a}$ at index i and $\boldsymbol{b}\{i; \boldsymbol{h}_{\text{disp}}[i]\}$ is referred to as the ith window of $\boldsymbol{b}$ with an offset of $\boldsymbol{h}_{\text{disp}}[i]$. The absolute value of $\boldsymbol{h}_{\text{disp}}[i]$ is the horizontal distance, denoted by $\boldsymbol{h}_{\text{dist}}[i]$.

Dynamic Window Matching (DWM) is a novel algorithm to find the corresponding windows between $\boldsymbol{a}$ and $\boldsymbol{b}$. The core of DWM is to determine the best $\boldsymbol{h}_{\text{disp}}$ such that $\boldsymbol{a}\{i\}$ corresponds to $\boldsymbol{b}\{i; \boldsymbol{h}_{\text{disp}}[i]\}$.

A Basic Algorithm to Find $\boldsymbol{h}_{\text{disp}}$. We present a basic algorithm to find $\boldsymbol{h}_{\text{disp}}$. For each window index i, we look for $\boldsymbol{a}\{i\}$ in the vicinity of $\boldsymbol{b}\{i\}$, as shown in Fig. 4. To be more precise, we perform TDE to detect $\boldsymbol{a}\{i\}$ in

$$\boldsymbol{b}\{i\}_{\text{E}} = \boldsymbol{b}[i \cdot n_{\text{hop}} - n_{\text{ext}} : i \cdot n_{\text{hop}} + n_{\text{ext}} + n_{\text{win}}], \tag{9}$$

where n_{ext} is a new parameter, called the extended window size, and $\boldsymbol{b}\{i\}_{\text{E}}$ is the extended ith window of $\boldsymbol{b}$. Suppose TDE returns a time delay of j. As a result, TDE thinks that $\boldsymbol{a}\{i\}$ is aligned with $\boldsymbol{b}\{i; j - n_{\text{ext}}\}$. We then let

$$\boldsymbol{h}_{\text{disp}}[i] = j - n_{\text{ext}}. \tag{10}$$

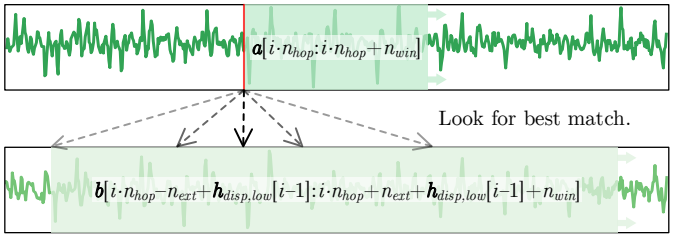


Fig. 4: Illustration of the DWM algorithm. The abscissas are time whereas the ordinates are signal values. A pair of sliding windows are established on the signals to be compared. As the windows slide across the signals, Time Delay Estimation (TDE) is used to determine the relative timing relationship between the pair of windows.

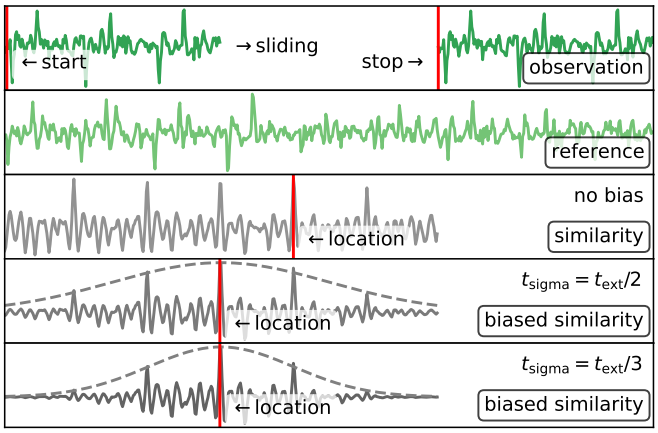


Fig. 5: Illustration of Time Delay Estimation with Bias (TDEB). For each subfigure, the abscissa is time and the ordinate is explained in the lower right box.

Extending the Range of $\boldsymbol{h}_{\text{disp}}$. The basic algorithm to find $\boldsymbol{h}_{\text{disp}}[i]$ might work well if the magnitude of the actual $\boldsymbol{h}_{\text{disp}}[i]$ does not exceed n_{ext}. Otherwise, the algorithm will definitely fail. This is because, according to Eq. (10), the range of $\boldsymbol{h}_{\text{disp}}[i]$ is $[-n_{\text{ext}}, n_{\text{ext}}]$. Hence, if the magnitude of the actual $\boldsymbol{h}_{\text{disp}}[i]$ exceeds n_{ext}, it is impossible for the basic algorithm to return a correct $\boldsymbol{h}_{\text{disp}}[i]$.

To solve this problem, we perform TDE to detect $\boldsymbol{a}\{i\}$ in $\boldsymbol{b}\{i; \boldsymbol{h}_{\text{disp}}[i-1]\}_{\text{E}}$. Suppose TDE returns j. We then perform the assignment

$$\boldsymbol{h}_{\text{disp}}[i] = j - n_{\text{ext}} + \boldsymbol{h}_{\text{disp}}[i-1]. \tag{11}$$

For $i = 0$, we define $\boldsymbol{h}_{\text{disp}}[i-1]$ to be 0.

Time Delay Estimation with Bias (TDEB). As shown in Fig. 5, when $\boldsymbol{a}\{i\}$ is mainly composed of periodic signals, multiple time delays could be returned by TDE with equal probability. Similarly, when $\boldsymbol{a}\{i\}$ is mainly composed of noise, TDE returns a random time delay. In a word, when $\boldsymbol{a}\{i\}$ is periodic or noisy, TDE is unstable.

To solve this problem, we rely on the assumption that $\boldsymbol{h}_{\text{disp}}[i]$ should be close to $\boldsymbol{h}_{\text{disp}}[i-1]$ most of the time. In other words, j should be close to n_{ext} most of the time.

When performing TDE, as an intermediate step, we obtain a similarity array $\boldsymbol{s}[j]$, where $j = 0, 1, \cdots, 2n_{\text{ext}} - 1$. To

increase similarity scores near $j = n_{\text{ext}}$, we multiply the similarity array by a Gaussian window with a standard deviation of n_{sigma} and a length of $2n_{\text{ext}}$, as shown in Fig. 5. We then continue TDE with the modified similarity array. In this way, we introduce bias towards $j = n_{\text{ext}}$. When $\boldsymbol{a}\{i\}$ is periodic or noisy, $\boldsymbol{h}_{\text{disp}}[i]$ will be close to $\boldsymbol{h}_{\text{disp}}[i-1]$.

Low Frequency Component of $\boldsymbol{h}_{\text{disp}}$. There is a new problem after extending the range of $\boldsymbol{h}_{\text{disp}}$. If, for any reason, the estimated value of $\boldsymbol{h}_{\text{disp}}[i-1]$ deviates significantly from its true value, it might cause $\boldsymbol{h}_{\text{disp}}[i]$ to deviate significantly from its true value, which in turn causes further deviation in $\boldsymbol{h}_{\text{disp}}[i+1]$, etc. In a word, the DWM process could run away. To mitigate this problem, we obtain a low frequency component of $\boldsymbol{h}_{\text{disp}}$ in the following way

$$\boldsymbol{h}_{\text{disp,low}}[i] = \text{round}(\eta(j - n_{\text{ext}}) + \boldsymbol{h}_{\text{disp,low}}[i-1]), \quad (12)$$

where η is a parameter that controls how fast $\boldsymbol{h}_{\text{disp,low}}$ can be affected by $j - n_{\text{ext}}$. Now, we perform TDEB to detect $\boldsymbol{a}\{i\}$ in $\boldsymbol{b}\{i; \boldsymbol{h}_{\text{disp,low}}[i-1]\}_{\text{E}}$. Suppose TDEB returns j. We then perform the assignment

$$\boldsymbol{h}_{\text{disp}}[i] = j - n_{\text{ext}} + \boldsymbol{h}_{\text{disp,low}}[i-1]. \quad (13)$$

The Final Algorithm to Find $\boldsymbol{h}_{\text{disp}}$. The final DWM algorithm is listed below, where $\text{TDEB}[\beta](\boldsymbol{x}, \boldsymbol{y})$ is a function that finds the time delay of $\boldsymbol{y}$ in $\boldsymbol{x}$ biased by a Gaussian window with a standard deviation of n_{sigma}.

Input: $\boldsymbol{a}$, $\boldsymbol{b}$, n_{win}, n_{hop}, n_{ext}, n_{sigma}, η
Output: $\boldsymbol{h}_{\text{disp}}[i], i = 0, 1, \cdots$

Define $\boldsymbol{h}_{\text{disp}}$ as a vector that can increase in size.
Define $\boldsymbol{h}_{\text{disp,low}}$ as a vector that can increase in size.
Add a special element $\boldsymbol{h}_{\text{disp,low}}[-1] = 0$.
$i = 0$
Wait for the printing process to start.
while the printing process is not over **do**
  Wait for $\boldsymbol{a}[i \cdot n_{\text{hop}} : i \cdot n_{\text{hop}} + n_{\text{win}}]$ to be available.
  j = $\text{TDEB}[n_{\text{sigma}}]($
    $\boldsymbol{b}[i \cdot n_{\text{hop}} - n_{\text{ext}} + \boldsymbol{h}_{\text{disp,low}}[i-1] :$
      $i \cdot n_{\text{hop}} + n_{\text{ext}} + \boldsymbol{h}_{\text{disp,low}}[i-1] + n_{\text{win}}]$,
    $\boldsymbol{a}[i \cdot n_{\text{hop}} : i \cdot n_{\text{hop}} + n_{\text{win}}]$
)
  $\boldsymbol{h}_{\text{disp}}[i] = j - n_{\text{ext}} + \boldsymbol{h}_{\text{disp,low}}[i-1]$
  $\boldsymbol{h}_{\text{disp,low}}[i] = \text{round}(\eta(j - n_{\text{ext}}) + \boldsymbol{h}_{\text{disp,low}}[i-1])$
  $i = i + 1$
end while
return $\boldsymbol{h}_{\text{disp}}$

C. Parameters in DWM

In this section, we explore how the parameters in DWM (n_{win}, n_{hop}, n_{ext}, n_{sigma}, and η) affect the performance of DWM and how each parameter should be selected for the best performance. The five parameters are defined in terms of indexes. The five parameters can also be defined in terms of seconds. We define $t_{\text{win}} = n_{\text{win}}/f_s$, $t_{\text{hop}} = n_{\text{hop}}/f_s$, $t_{\text{ext}} = n_{\text{ext}}/f_s$, and $t_{\text{sigma}} = n_{\text{sigma}}/f_s$, where f_s is the sampling rate of the side-channel signal.

Parameters t_{ext} and t_{sigma}. As shown in Fig. 5, the ratio $t_{\text{ext}}/t_{\text{sigma}}$ controls the strength of the bias in TDEB, and a higher value of $t_{\text{ext}}/t_{\text{sigma}}$ corresponds to a stronger bias towards the center. When $t_{\text{ext}}/t_{\text{sigma}} < 1$, the bias effect is not significant. When $t_{\text{ext}}/t_{\text{sigma}} > 1$, the bias effect is significant and the extended window size is effectively determined by t_{sigma} instead of t_{ext}. By default, we use $t_{\text{ext}}/t_{\text{sigma}} = 2$ for two reasons. First, bias is desirable. Hence $t_{\text{ext}}/t_{\text{sigma}} > 1$. Second, when $t_{\text{ext}}/t_{\text{sigma}} > 3$, to maintain the same effective extended window size, increasing the ratio is tantamount to increasing t_{ext}. This merely increases the consumption of computational resources without other effects. As a balance, we choose $t_{\text{ext}}/t_{\text{sigma}} = 2$.

With $t_{\text{ext}}/t_{\text{sigma}} = 2$, t_{sigma} effectively determines the extended window size. The influence of t_{sigma} on $\boldsymbol{h}_{\text{disp}}$ is shown in Fig. 6 (a). To ensure a successful DWM process, t_{sigma} should be larger than the absolute difference of the actual $\boldsymbol{h}_{\text{disp}}$ between any two consecutive windows. At the same time, t_{sigma} should not be too large, as it not only requires more computational resources but also decreases the accuracy of DWM as a wider search area has more distraction.

To select the best t_{sigma}, we start with a large t_{sigma} and obtain the maximum value of the absolute difference of $\boldsymbol{h}_{\text{disp}}$ between any two consecutive windows. We select t_{sigma} to be a value that is larger than this maximum value.

Parameter t_{hop}. t_{hop} controls the temporal resolution of $\boldsymbol{h}_{\text{disp}}$. The maximum value of t_{hop} is t_{win} whereas the minimum value of t_{hop} is $1/f_s$. It is desirable to have a higher resolution by choosing a smaller t_{hop}. However, the computational cost increases significantly as t_{hop} is reduced. As a balance between computational cost and temporal resolution, we choose $t_{\text{hop}} = t_{\text{win}}/2$ by default.

Parameter t_{win}. t_{win} is the window size in the TDE process. Fig. 6 (b) shows how t_{win} affects $\boldsymbol{h}_{\text{disp}}$. When t_{win} is very small, there are a lot of spikes in $\boldsymbol{h}_{\text{disp}}$. When t_{win} is very large, the temporal resolution of $\boldsymbol{h}_{\text{disp}}$ becomes lower.

In NSYNC, we obtain the best t_{win} by parametric analysis. We sweep t_{win} from a small value to a large value and select the t_{win} such that the change of the overall shape of $\boldsymbol{h}_{\text{disp}}$ is the smallest with respect to t_{win}.

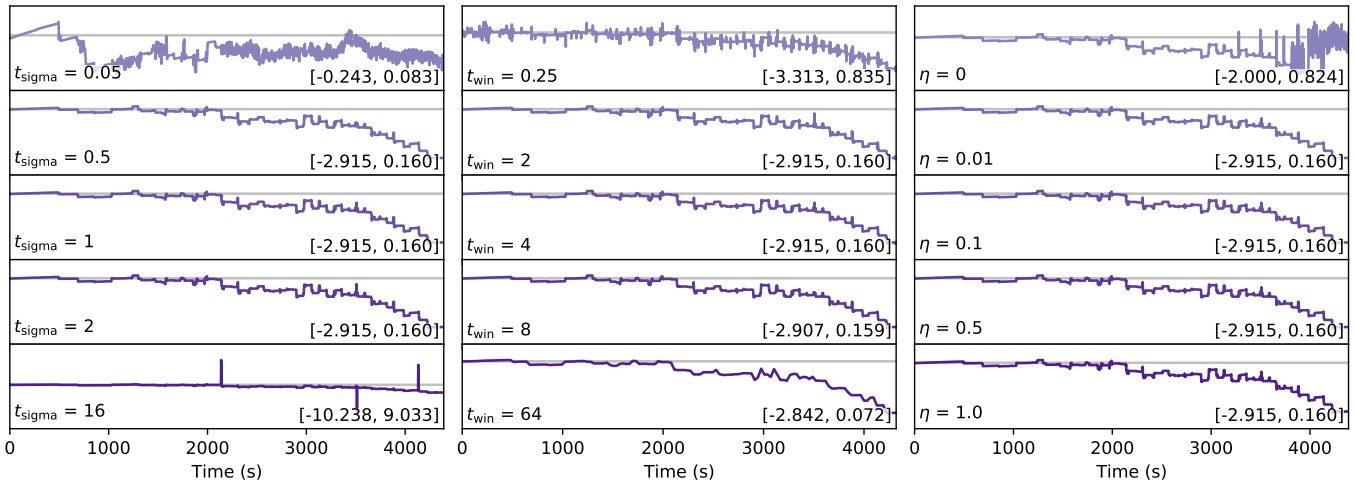
Parameter η. Fig. 6 (c) shows how η affects $\boldsymbol{h}_{\text{disp}}$. In general, it is necessary to have a positive η. For rare situations, when η is close to 1.0, DWM could run away.

To select the best η, we start with a small value of η, typically $\eta = 0.1$. If DWM is unable to converge, we can crank up this value until DWM converges.

VII. Design of the NSYNC Framework

NSYNC is a framework of IDSs, and any IDS that conforms to the general structure outlined in Fig. 7 can be considered an instance of the NSYNC framework.

Overall Structure of NSYNC. Suppose the observed side-channel signal $\boldsymbol{a}$ and the reference signal $\boldsymbol{b}$ are aligned at the beginning of their printing processes. As shown in Fig.



(a) Parametric analysis of t_{sigma}. (b) Parametric analysis of t_{win}. (c) Parametric analysis of η.

Fig. 6: Parametric analysis of t_{sigma}, t_{win}, and η. The ordinates are $\boldsymbol{h}_{\text{disp}}$. The brackets in all figures show the range of $\boldsymbol{h}_{\text{disp}}$.

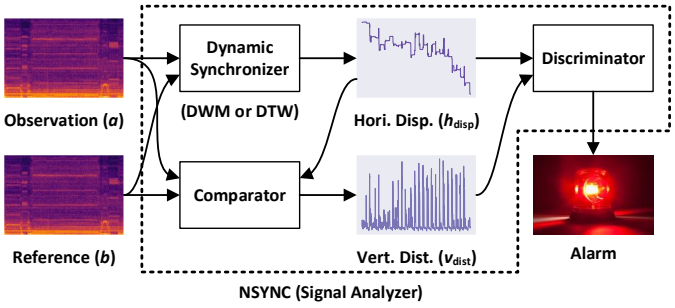


Fig. 7: Overview of the NSYNC framework.

7, in an NSYNC IDS, $\boldsymbol{a}$ and $\boldsymbol{b}$ first goes through a dynamic synchronizer to obtain the horizontal displacement array $\boldsymbol{h}_{\text{disp}}$, which describes the corresponding points or windows between $\boldsymbol{a}$ and $\boldsymbol{b}$. Afterwards, $\boldsymbol{a}$ and $\boldsymbol{b}$ goes through a comparator to obtain the vertical distance array $\boldsymbol{v}_{\text{dist}}$. Finally, both $\boldsymbol{h}_{\text{disp}}$ and $\boldsymbol{v}_{\text{dist}}$ are used by a discriminator to determine if $\boldsymbol{a}$ and $\boldsymbol{b}$ are significantly different at any moment. If so, an alert is issued.

In this following sections, we describe the components in NSYNC respectively. The first component, the dynamic synchronizer, has already been described in Section VI.

A. Comparator: Vertical Distance Calculation

Once the corresponding points or windows between $\boldsymbol{a}$ and $\boldsymbol{b}$ are identified, a distance value can be calculated for each pair of points or windows, which are further used by the discriminator for intrusion detection.

Distance Metrics. Suppose $\boldsymbol{u}$ and $\boldsymbol{v}$ are 1-D vectors of the same length N. One distance metric is the correlation distance

$$d(\boldsymbol{u}, \boldsymbol{v}) = 1 - \frac{(\boldsymbol{u} - \mu_u) \cdot (\boldsymbol{v} - \mu_v)}{\|\boldsymbol{u} - \mu_u\| \cdot \|\boldsymbol{v} - \mu_v\|}. \tag{14}$$

where $\|\cdot\|_2$ is the L2 norm operator and μ_u and μ_v are defined in Eqn. (4). The second term in Eq. (14) is in fact the Pearson's correlation coefficient between $\boldsymbol{u}$ and $\boldsymbol{v}$.

When $\boldsymbol{u}$ and $\boldsymbol{v}$ are 2-D vectors with the same length N and the same number of channels C, as with the similarity function in Section V-B, we can calculate the distance metric along the time axis for each channel and then average the distance metrics across the channels.

In NSYNC, we use the correlation distance by default. There are many other distance metrics, such as the Manhattan distance and the Euclidean distance. However, we do not consider the two distance metrics because they are sensitive to the overall amplitudes (aka the gains) of $\boldsymbol{a}$ and $\boldsymbol{b}$, and the gains of many side-channel signals are susceptible to changes.[2] If a distance metric that is sensitive to the gains of $\boldsymbol{a}$ and $\boldsymbol{b}$ is used in an IDS, the gains of $\boldsymbol{a}$ and $\boldsymbol{b}$ must be strictly controlled (which can be very hard). Otherwise, the IDS will suffer from a lot of false alerts.

Vertical Distance $\boldsymbol{v}_{\text{dist}}$. If $\boldsymbol{a}$ and $\boldsymbol{b}$ are synchronized by DTW, we can calculate $\boldsymbol{v}_{\text{dist}}$ between $\boldsymbol{a}$ and $\boldsymbol{b}$ point by point. Suppose (i, j) is a tuple returned by DTW, we define $\boldsymbol{v}_{\text{dist}}[i] = d(\boldsymbol{a}[i], \boldsymbol{b}[j])$. If there are multiple tuples with the first index being i, such as $(i, j_1), (i, j_2), \cdots, (i, j_{K_i})$, we define

$$\boldsymbol{v}_{\text{dist}}[i] = \frac{1}{K_i} \sum_{k=1}^{K_i} d(\boldsymbol{a}[i], \boldsymbol{b}[j_k]). \tag{15}$$

If $\boldsymbol{a}$ and $\boldsymbol{b}$ are synchronized by DWM, we can calculate the vertical distances between $\boldsymbol{a}$ and $\boldsymbol{b}$ window by window. Suppose $\boldsymbol{a}\{i\}$ is the ith window of $\boldsymbol{a}$ and its corresponding window in $\boldsymbol{b}$ is $\boldsymbol{b}\{i; \boldsymbol{h}_{\text{disp}}[i]\}$. We have

$$\boldsymbol{v}_{\text{dist}}[i] = d(\boldsymbol{a}\{i\}, \boldsymbol{b}\{i; \boldsymbol{h}_{\text{disp}}[i]\}). \tag{16}$$

B. Discriminator: Automatic Intrusion Detection

The discriminator checks $\boldsymbol{a}$ and $\boldsymbol{b}$ in real time and automatically determines if $\boldsymbol{a}$ is significantly different from $\boldsymbol{b}$. If so, an

[2]For example, the amplitude of the acoustic side-channel signal strongly depends on the distance from the microphone to the printer as well as the gain of the ADC converter, both of which are susceptible to changes.

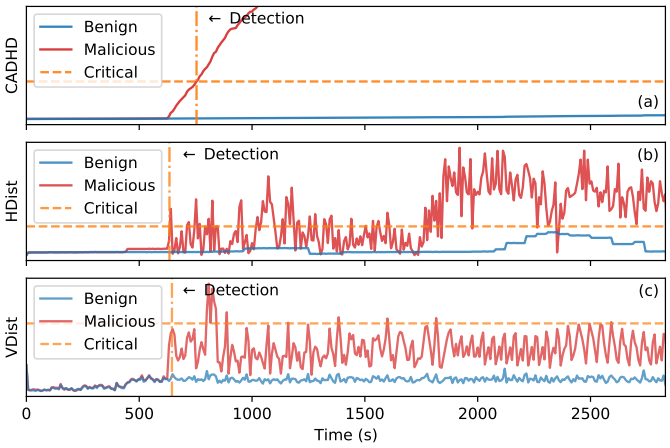


Fig. 8: Automatic Intrusion Detection. (a) Intrusion is detected if $\boldsymbol{c}_{\text{disp}}[i]$ exceeds c_c. (b) Intrusion is detected if $\boldsymbol{h}_{\text{dist}}[i]$ exceeds h_c. (c) Intrusion is detected if $\boldsymbol{v}_{\text{dist}}[i]$ exceeds v_c.

intrusion is declared. The discriminator is composed of three sub-modules. If any sub-module raises an alert, an intrusion is declared. Each sub-module is discussed as follows.

Sub-Module 1: $\boldsymbol{c}_{\text{disp}}$-Based Detection. This sub-module checks $\boldsymbol{h}_{\text{disp}}$ to determine if $\boldsymbol{a}$ and $\boldsymbol{b}$ are successfully synchronized. When DSYNC succeeds, $\boldsymbol{h}_{\text{disp}}$ contains a few fluctuations, such as the benign process in Fig. 8 (b). In contrast, when DSYNC fails, $\boldsymbol{h}_{\text{disp}}$ contains a lot of fluctuations, such as the malicious process in Fig. 8 (b). To capture this feature, we calculate the Cumulative Absolute Difference of the Horizontal Displacement (CADHD)

$$\boldsymbol{c}_{\text{disp}}[i] = \sum_{j=0}^{i} |\boldsymbol{h}_{\text{disp}}[j] - \boldsymbol{h}_{\text{disp}}[j-1]|, \tag{17}$$

where $\boldsymbol{h}_{\text{disp}}[-1]$ is defined to be zero. Fig. 8 (a) shows the CADHD arrays for a benign process where DSYNC succeeded and a malicious process where DSYNC failed. An intrusion is detected at index i if

$$\boldsymbol{c}_{\text{disp}}[i] > c_c, \tag{18}$$

where c_c is a critical value to be determined later.

Sub-Module 2: $\boldsymbol{h}_{\text{dist}}$-Based Detection. For $\boldsymbol{h}_{\text{dist}}$, as shown in Fig. 8 (b), an intrusion is detected at index i if

$$\boldsymbol{h}_{\text{dist}}[i] > h_c, \tag{19}$$

where h_c is a critical horizontal distance to be determined.

Sub-Module 3: $\boldsymbol{v}_{\text{dist}}$-Based Detection. For $\boldsymbol{v}_{\text{dist}}$, as shown in Fig. 8 (c), an intrusion is detected at index i if

$$\boldsymbol{v}_{\text{dist}}[i] > v_c, \tag{20}$$

where v_c is a critical vertical distance to be determined.

Suppressing Spikes. There are spikes in $\boldsymbol{h}_{\text{dist}}$ and $\boldsymbol{v}_{\text{dist}}$ due to time noise and amplitude noise. The spikes could cause false positives. To mitigate this problem, we filter $\boldsymbol{h}_{\text{dist}}$ and $\boldsymbol{v}_{\text{dist}}$ in the following ways before Eq. (19) and Eq. (20) are applied:

$$\boldsymbol{h}_{\text{dist},f}[i] = \min(\boldsymbol{h}_{\text{dist}}[i-n:i]), \quad i = 0, 1, \cdots, \tag{21}$$

$$\boldsymbol{v}_{\text{dist},f}[i] = \min(\boldsymbol{v}_{\text{dist}}[i-n:i]), \quad i = 0, 1, \cdots, \tag{22}$$

where $\boldsymbol{h}_{\text{dist},f}$ and $\boldsymbol{v}_{\text{dist},f}$ are filtered arrays and n is the window size of the filter. By default, we use a window size of 3 for both $\boldsymbol{h}_{\text{dist}}$ and $\boldsymbol{v}_{\text{dist}}$.

C. Learning Critical Values for the Discriminator

In this section, we describe a One-Class Classification (OCC) scheme to determine the critical values c_c, h_c, and v_c in the discriminator. For this purpose, for one reference signal $\boldsymbol{b}$, we need to run the benign process M times and observe the side-channel signals $\boldsymbol{a}_m$, $m = 0, 1, \cdots, M-1$, where M is the number of observed signals.

Suppose $\boldsymbol{c}_{\text{disp,m}}$, $\boldsymbol{h}_{\text{dist,m}}$ and $\boldsymbol{v}_{\text{dist,m}}$ are obtained by comparing $\boldsymbol{a}_m$ and $\boldsymbol{b}$. Here, we assume that $\boldsymbol{h}_{\text{dist,m}}$ and $\boldsymbol{v}_{\text{dist,m}}$ are the **filtered** horizontal distance array the **filtered** vertical distance array respectively. We have

$$c_{c,m} = \max_i \boldsymbol{c}_{\text{disp,m}}[i], \tag{23}$$

$$h_{c,m} = \max_i \boldsymbol{h}_{\text{dist,m}}[i], \tag{24}$$

$$v_{c,m} = \max_i \boldsymbol{v}_{\text{dist,m}}[i]. \tag{25}$$

The critical distances are determined by

$$c_c = \max_m c_{c,m} + r\left(\max_m c_{c,m} - \min_m c_{c,m}\right), \tag{26}$$

$$h_c = \max_m h_{h,m} + r\left(\max_m h_{h,m} - \min_m h_{h,m}\right), \tag{27}$$

$$v_c = \max_m v_{v,m} + r\left(\max_m v_{v,m} - \min_m v_{v,m}\right), \tag{28}$$

The parameter r determines the False Positive Rate (FPR) and the False Negative Rate (FNR). The higher the value of r, the lower the FPR, but the higher the FNR. The value of r depends on M, the sample size. To maintain the same FPR, r gets smaller when M becomes larger. In NSYNC, we select an r that results in a small FPR (< 0.05) for most scenarios.

VIII. Evaluation

This section describes experiments to evaluate the performance of NSYNC and existing IDSs.

A. Experiment Setup

Printers. We performed experiments on an Ultimaker 3 printer (UM3) and a SeeMeCNC Rostock Max V3 printer (RM3). The UM3 printer is the most popular desktop 3D printer [2], whereas the RM3 printer is a popular Delta printer.

Printing Processes. We selected a gear model with a diameter of 60 mm and a thickness of 7.5 mm. For UM3, we used Cura 4.4 as the slicer. For RM3, we used MatterControl 1.7.5 with MatterSlice as the slicer. For both printers, we used the default setting with a layer height of 0.2 mm.

For each printer, the benign process was repeated 151 times. One benign process served as the reference. 50 benign processes were used for learning the critical values. The other

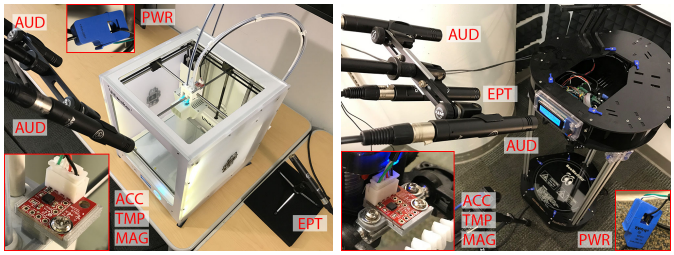


Fig. 9: Experimental setups. (a) Ultimaker 3 and various sensors, (b) SeeMeCNC Rostock Max V3 and various sensors.

100 benign processes were used for testing. We manipulated the benign G-code file in five ways to simulate five types of malicious printing processes in the literature. All malicious processes were used for testing. The details of the printing processes for each printer can be found in Table I.

TABLE I: Printing Processes for Each Printer

B/M	Process	Re.	Description	Ref.
B	Benign	1	This is used for the reference.	
B	Benign	50	These are used for training.	
B	Benign	100	These are used for testing.	
M	Void	20	A void is inserted.	[25]
M	InfillGrid	20	Infill pattern is changed to grid.	[4]
M	Speed0.95	20	Printing speed is decreased by 5%.	[12]
M	Layer0.3	20	Layer height is changed to 0.3 mm.	[12]
M	Scale0.95	20	The object is shrunk by 5%.	[25]

B/M = Benign or Malicious. Re. = Repetition (for each printer). All the malicious processes are used for testing.

Side Channels. We used six different types of side channels in the experiments and the details are in Table II. The locations of the sensors are shown in Fig. 9. We installed the MPU9250 sensors on the printheads of the printers. SCT013 measured the total AC currents delivered to the printers.

TABLE II: Types of Side Channels

ID	Side Channel	Sensor	*fs* (Hz)	CHs	Bits
ACC	Acceleration	MPU9250	4000	6	16
TMP	Temperature	MPU9250	4000	1	16
MAG	Magnetic	MPU9250	100	3	16
AUD	Audio	AKG170	48,000	2	24
EPT	Elec. Potential	AKG170*	96,000	1	24
PWR	Power/Current	SCT013	12,000	1	24

EPT = Electric Potentials. CHs = Number of Channels. * The AKG170 for collecting electric potentials was modified by removing the cap, inspired by the method in [14].

Spectrograms. Many existing IDSs internally transform a side-channel signal into a spectrogram before further processing [4], [5], [13]. For other IDSs, including NSYNC, in addition to comparing raw signals directly, we also compared their spectrograms. For each side-channel signal, we obtained its spectrogram via Short-Time Fourier Transforms (STFT) [17] and the details are shown in Table III. The spectrogram of a signal can be considered a new signal with a reduced sampling rate and an increased number of channels.

TABLE III: Spectrograms for Side Channels

ID	Δf (Hz)	Δt (s)	Window	CHs	Bits
ACC	20	1/80	BH	101×6	16
TMP	20	1/80	BH	101	16
MAG	5	1/20	BH	11×3	16
AUD	120	1/240	BH	201×2	16
EPT	120	1/240	BH	401	16
PWR	60	1/120	Boxcar	101	16

Δf is the spectral resolution, which is equal to the reciprocal of the window size (in seconds) in STFT. Δt is the temporal resolution, which is equal to the time by which the window moves forward each time in STFT. BH = Blackman-Harris.

Parameters for DWM. We selected the DWM parameters according to the methods outlined in Section VI-C. The parameters for UM3 and RM3 are listed in Table IV. These parameters work well for a variety of side channels and printing processes.

TABLE IV: Parameters in DWM

Printer	t_{win}	t_{hop}	t_{ext}	t_{sigma}	η
UM3	4.0 s	2.0 s	2.0 s	1.0 s	0.1
RM3	1.0 s	0.5 s	0.1 s	0.05 s	0.1

B. Consistency of Horizontal Displacements

Fig. 10 shows $\boldsymbol{h}_{\text{disp}}$ obtained by six different side channels and two different transformations (raw signals or spectrograms) for a particular benign process. We can see that $\boldsymbol{h}_{\text{disp}}$ obtained by ACC and AUD are almost identical, regardless of the transformation. Although $\boldsymbol{h}_{\text{disp}}$ obtained by the raw signal of EPT does not make sense, $\boldsymbol{h}_{\text{disp}}$ obtained by the spectrogram of EPT is almost identical to the $\boldsymbol{h}_{\text{disp}}$ obtained by ACC or AUD. In addition, although there appears to be a lot of noise in $\boldsymbol{h}_{\text{disp}}$ obtained by MAG for both transformations, the overall shape of the $\boldsymbol{h}_{\text{disp}}$ is the same as the overall shape of $\boldsymbol{h}_{\text{disp}}$ obtained by ACC or AUD. In contrast, $\boldsymbol{h}_{\text{disp}}$ obtained by TMP and PWR are noise like, regardless of the transformation.

The raw signal of EPT is mostly composed of a 60 Hz power component, which is not correlated with the state of the printer. In contrast, the spectrogram of EPT contains 401 channels and the 60 Hz power component is only one of them. As all channels are treated with the same level of importance, the influence of the 60 Hz power component is not prominent.

The consistency of $\boldsymbol{h}_{\text{disp}}$ across side-channel signals that are highly correlated with the state of the printer indicates that $\boldsymbol{h}_{\text{disp}}$ is a property of the printing process, not the side channels. In the rest of the paper, we drop TMP and PWR as they are weakly correlated with the state of the printer. In addition, we also drop the raw signal of EPT but keep the spectrogram of EPT.

C. Results for IDSs without DSYNC

This section presents evaluation results for IDSs which do not contain any form of DSYNC.

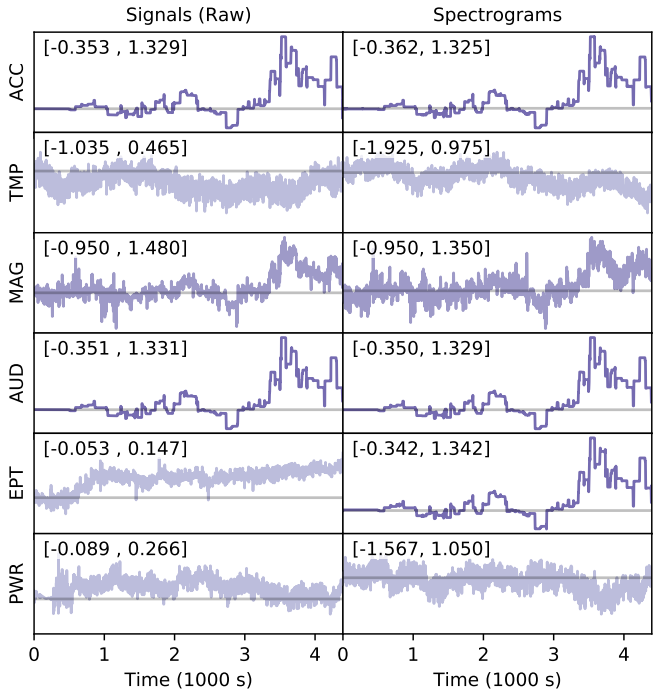


Fig. 10: $\boldsymbol{h}_{\text{disp}}$ obtained by six different side channels and two different transformations (raw signals and spectrograms).

TABLE V: Results for Moore's and Gao's IDSs

P	Side Ch.	Moore's Method		Gao's Method	
		Raw	Spectro.	Raw	Spectro.
UM3	ACC	0.05 / 0.01	0.01 / 0.00	0.01 / 0.02	0.03 / 0.03
	MAG	0.03 / 0.01	0.03 / 0.02	0.01 / 0.08	0.12 / 0.12
	AUD	0.05 / 0.01	0.05 / 0.05	0.05 / 0.02	0.05 / 0.05
	EPT	0.31 / 0.25	0.00 / 0.00	0.30 / 0.25	0.01 / 0.00
RM3	ACC	0.00 / 0.00	0.02 / 0.03	**0.01 / 1.00**	0.00 / 0.03
	MAG	0.08 / 0.15	0.03 / 0.07	0.07 / 0.15	0.10 / 0.08
	AUD	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.00 / 0.02
	EPT	0.18 / 0.21	0.09 / 0.36	0.31 / 0.37	0.00 / 0.13

P = Printer. Side Ch. = Side Channel. The result format is FPR / TPR, where FPR = False Positive Rate and TPR = True Positive Rate. We emboldened the results where the accuracy is at least 0.95. We grayed (and dropped) results for the raw signal of EPT.

Moore's IDS [18]. This IDS essentially compares $\boldsymbol{a}[n]$ and $\boldsymbol{b}[n]$ without DSYNC to obtain $\boldsymbol{v}_{\text{dist}}[n]$ for $n = 0, 1, \cdots$, where the distance metric is the Mean Absolute Error (MAE). The IDS is originally designed for electric currents in motors. However, we were not able to observe this type of side-channel signals. Instead, we applied this IDS on available side-channel signals. The results are shown in Table V.

Bayen's IDS [4]. This IDS compares side-channel signals window by window (90 s or 120 s for the window size). This IDS first checks if the windows are in sequence. If not, an intrusion is declared. It then checks the scores for each window. If the score of any window is below a pre-defined threshold, an intrusion is declared. However, there are no detail on how to obtain the thresholds for a new printer in [4]. As a result, we used the OCC method in NSYNC to determine the thresholds. We used $r = 0.0$ because the TPRs for the threshold-based sub-module are very low. We only tested this IDS on AUD as this IDS only supports the acoustic side channel. The results are shown in Table VI.

TABLE VI: Detection Results for Bayens' IDS

Printer	Window Size (s)	Overall Results	Sub-Module Results	
			Sequence	Threshold
UM3	90	1.00 / 1.00	1.00 / 1.00	0.18 / 0.31
	120	1.00 / 1.00	1.00 / 1.00	0.10 / 0.18
RM3	90	0.51 / 1.00	0.51 / 1.00	0.07 / 0.97
	120	0.30 / 1.00	0.29 / 1.00	0.04 / 0.63

Results are for AUD only. See Table V for more table notes.

Belikovetsky's IDS [5]. This IDS applies the Principle Component Analysis (PCA) to compress the number of channels of the spectrogram of the observed signal down to three. Suppose the result is $\boldsymbol{a}$. The reference signal goes through the same process. Suppose the result is $\boldsymbol{b}$. $\boldsymbol{a}$ and $\boldsymbol{b}$ are then compared point by point without DSYNC using the cosine distance metric [5]. Suppose the result is $\boldsymbol{v}_{\text{dist}}$. A window of five seconds is used to calculate the moving average of $\boldsymbol{v}_{\text{dist}}$. If the average distances of four consecutive windows drop below 0.63, then an intrusion is detected. As with the Bayen's IDS, this method only supports the acoustic side channel. The intrusion detection results are FPR / TPR = 1.00 / 1.00 for UM3 and FPR / TPR = 0.31 / 1.00 for RM3.

D. Results for IDSs with Coarse DSYNC

This section presents evaluation results for IDSs which align the signals at the beginning of each layer. Since a layer can be considered a huge window, this behavior can be considered as a form of DSYNC, but on a coarse level. Nevertheless, these IDSs are not aware of time noise.

Gao's IDS [12]. This IDS is similar to the Moore's IDS except two aspects. First, $\boldsymbol{a}$ and $\boldsymbol{b}$ are synchronized at moments when a layer change happens. Second, there is no discriminator in the Gao's IDS. As a result, we use the discriminator in NSYNC. We used $r = 0.0$ because the TPRs are very low. The results are shown in Table V.

Gatlin's IDS [13]. The details of this IDS can be found in [13] or Section III. This IDS is originally designed for the electric currents in motors and the layer changing moments are determined by detecting activities in the currents in the Z motor. Since we were not able to access the currents in any motor, we obtained the layer changing moments manually, and we applied this IDS to the side-channel signals that we obtained. The detection results are shown in Table VII.

E. Results for IDSs with Fine DSYNC

This section presents results for IDSs which dynamically synchronize signals to be compared on a fine scale. The IDSs in this section are fully aware of time noise.

NSYNC/DWM. We evaluated NSYNC with DWM as its dynamic synchronizer. We used $r = 0.3$ in the OCC training

TABLE VII: Detection Results for Gatlin's IDS

Printer	Side Ch.	Overall Results	Sub-Module Results	
			Time	Match
UM3	ACC	0.30 / 1.00	0.15 / 1.00	0.17 / 0.62
	MAG	0.53 / 1.00	0.16 / 1.00	0.44 / 0.38
	AUD	0.22 / 1.00	0.14 / 1.00	0.09 / 0.07
	EPT	**0.05** / **0.98**	0.05 / 0.98	0.00 / 0.02
RM3	ACC	0.29 / 1.00	0.07 / 1.00	0.26 / 1.00
	MAG	0.17 / 1.00	0.05 / 1.00	0.12 / 1.00
	AUD	0.20 / 1.00	0.10 / 1.00	0.11 / 1.00
	EPT	**0.08** / **1.00**	0.08 / 1.00	0.00 / 0.00

See Table V for table notes.

TABLE VIII: Detection Results for NSYNC with DWM

P	T	Side Ch.	Overall Results	Individual Sub-Module Results		
				c_{disp}	h_{dist}	v_{dist}
UM3	Raw	ACC	**0.02** / **1.00**	0.00 / 1.00	0.02 / 0.64	0.00 / 1.00
		MAG	**0.00** / **1.00**	0.00 / 1.00	0.00 / 0.93	0.00 / 0.51
		AUD	**0.02** / **1.00**	0.00 / 1.00	0.02 / 0.47	0.00 / 0.08
		EPT	0.00 / 0.06	0.00 / 0.06	0.00 / 0.00	0.00 / 0.06
	Spectro.	ACC	**0.02** / **1.00**	0.00 / 1.00	0.02 / 0.73	0.00 / 0.80
		MAG	**0.01** / **1.00**	0.00 / 1.00	0.01 / 0.87	0.00 / 0.56
		AUD	**0.02** / **1.00**	0.00 / 1.00	0.02 / 0.83	0.00 / 1.00
		EPT	**0.00** / **1.00**	0.00 / 1.00	0.00 / 0.52	0.00 / 1.00
RM3	Raw	ACC	**0.00** / **1.00**	0.00 / 1.00	0.00 / 0.80	0.00 / 1.00
		MAG	**0.01** / **1.00**	0.01 / 1.00	0.00 / 1.00	0.00 / 1.00
		AUD	**0.00** / **1.00**	0.00 / 1.00	0.00 / 0.57	0.00 / 1.00
		EPT	0.00 / 0.21	0.00 / 0.05	0.00 / 0.00	0.00 / 0.21
	Spectro.	ACC	**0.00** / **1.00**	0.00 / 1.00	0.00 / 0.91	0.00 / 0.03
		MAG	**0.00** / **1.00**	0.00 / 1.00	0.00 / 0.00	0.00 / 0.00
		AUD	**0.00** / **1.00**	0.00 / 1.00	0.00 / 0.91	0.00 / 1.00
		EPT	**0.00** / **1.00**	0.00 / 1.00	0.00 / 0.63	0.00 / 0.00

T = Transform (on Signals). See Table V for more table notes.

process to bring down the overall FPR close to zero. The detection results are shown in Table VIII. The column c_{disp} shows the results if CADHD is used alone for intrusion detection. Similarly, the columns h_{dist} and v_{dist} show the results if h_{dist} and v_{dist} are used alone respectively.

NSYNC/DTW. We evaluated NSYNC with DTW as its dynamic synchronizer. We used $r = 0.3$ to bring down the overall FPR close to zero. We were not able to apply DTW on the raw signals because it took forever for DTW to synchronize them. The detection results are shown in Table IX.

In addition, we measured the average time it took to analyze one second of the spectrograms of the side-channel signals for both DWM and DTW. The results are shown in Fig. 11. We can see that DTW is much slower than DWM, even if we used FastDTW with the fastest configuration to implement DTW.

F. Overall Comparison between IDSs

We define the accuracy of an IDS as the number of correctly identified processes to the number of total processes. Since in our experiments the number of benign processes is equal to the number of malicious processes, we can calculate the accuracy by $[(1 - \text{FPR}) + \text{TPR}]/2$.

TABLE IX: Detection Results for NSYNC with DTW

P	T	Side Ch.	Overall Results	Individual Sub-Module Results		
				c_{disp}	h_{dist}	v_{dist}
UM3	Spectro.	ACC	**0.02** / **1.00**	0.02 / 1.00	0.00 / 1.00	0.00 / 0.00
		MAG	0.10 / 0.26	0.08 / 0.04	0.10 / 0.24	0.00 / 0.00
		AUD	**0.06** / **1.00**	0.06 / 1.00	0.06 / 1.00	0.00 / 0.00
		EPT	0.04 / 0.24	0.00 / 0.00	0.00 / 0.22	0.04 / 0.04
RM3	Spectro.	ACC	0.02 / 0.40	0.02 / 0.40	0.02 / 0.40	0.00 / 0.00
		MAG	0.00 / 0.40	0.00 / 0.40	0.00 / 0.40	0.00 / 0.00
		AUD	**0.00** / **1.00**	0.00 / 0.90	0.00 / 1.00	0.00 / 0.00
		EPT	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00

T = Transform (on Signals). See Table V for more table notes.

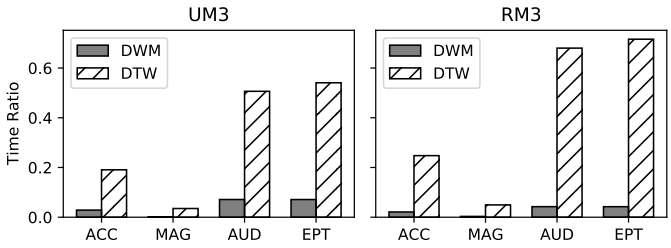


Fig. 11: Average time it took for DWM and DTW to dynamically synchronize one second of the spectrograms of the side-channel signals. This is also known as the time ratio. The results were averaged over the side channels.

Figs. 12 show the average accuracy for all IDSs that we evaluated. The results were averaged over all printers, all types of side channels, and all transformations (except the raw signal of EPT). As we can see in Fig. 12, as the level of DSYNC increases from none to fine, the overall accuracy of the IDSs increases. Any IDS labeled with the symbol "T" in Fig. 12 means that the IDS uses time as an indicator for intrusion detection. By analyzing the results of sub-modules in Tables VIII and IX, we can see that time is a more effective indicator than amplitude for intrusion detection.

IX. Conclusion

This paper presents NSYNC, a practical framework of IDSs that leverage side-channel signals in AM systems. The primary motivation to propose NSYNC is that we noticed the existence of time noise in AM printing processes, which can invalidate existing IDSs. In addition, many existing IDSs require full knowledge of malicious printing processes in advance, which can be impractical. NSYNC addresses the time noise problem by using Dynamic SYNChronization (DSYNC) and uses One Class-Classification (OCC) for automatic intrusion detection, which does not require knowledge of malicious printing processes at all.

An existing method to perform DSYNC is Dynamic Time Warping (DTW). However, our evaluation results show that DTW is not suitable for analyzing side-channel signals in AM systems because DTW not only has limited accuracy but also consumes an excessive amount of computational resources. Our newly proposed method, Dynamic Window Matching (DWM), can successfully overcome the problems of DTW.

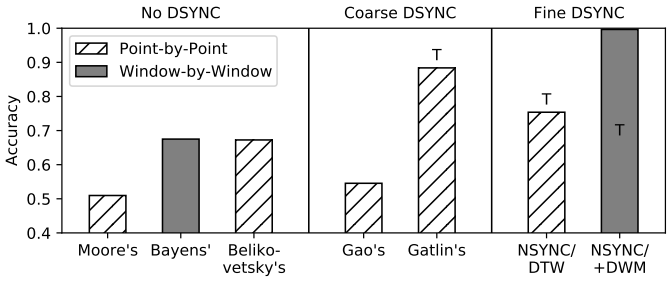


Fig. 12: Average accuracy of seven different IDSs in our evaluation. The symbol T means that the IDS uses time as an indicator for intrusion detection.

In the experiments we performed, NSYNC/DWM can achieve an accuracy of 0.99, beating NSYNC/DTW and all existing IDSs that use side-channel signals in AM systems.

ACKNOWLEDGMENT

This project is supported by National Science Foundation (NSF) under Grant CPS-1739259 and CPS-1931977.

REFERENCES

[1] M. A. Al Faruque, S. R. Chhetri, A. Canedo, and J. Wan, "Acoustic side-channel attacks on additive manufacturing systems," in *Proceedings of the 7th International Conference on Cyber-Physical Systems*, ser. ICCPS '16. Piscataway, NJ, USA: IEEE Press, 2016, pp. 19:1–19:10. [Online]. Available: http://dl.acm.org/citation.cfm?id=2984464.2984483

[2] T. Alsop, "3d printer market distribution," 2017. [Online]. Available: https://www.statista.com/statistics/756606/worldwide-3d-printer-manufacturer-market-distribution/

[3] ASTM, "Standard terminology for additive manufacturing – general principles – terminology," ASTM International, West Conshohocken, PA, Standard ISO/ASTM 52900:2015(E), 2015.

[4] C. Bayens, T. Le, L. Garcia, R. Beyah, M. Javanmard, and S. Zonouz, "See no evil, hear no evil, feel no evil, print no evil? malicious fill patterns detection in additive manufacturing," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, 2017, pp. 1181–1198. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/bayens

[5] S. Belikovetsky, Y. Solewicz, M. Yampolskiy, J. Toh, and Y. Elovici, "Digital audio signature for 3d printing integrity," *IEEE Transactions on Information Forensics and Security*, pp. 1–1, 2018.

[6] S. Belikovetsky, M. Yampolskiy, J. Toh, J. Gatlin, and Y. Elovici, "dr0wned – cyber-physical attack with additive manufacturing," in *11th USENIX Workshop on Offensive Technologies (WOOT 17)*. Vancouver, BC: USENIX Association, 2017. [Online]. Available: https://www.usenix.org/conference/woot17/workshop-program/presentation/belikovetsky

[7] J. Benesty, Y. Huang, and J. Chen, "Time delay estimation via minimum entropy," *IEEE Signal Processing Letters*, vol. 14, no. 3, pp. 157–160, 2007.

[8] S. Bhasin, A. Chattopadhyay, A. Heuser, D. Jap, S. Picek, and R. R. Shrivastwa, "Mind the portability: A warriors guide through realistic profiled side-channel analysis," *IACR Cryptology ePrint Archive*, vol. 2019, p. 661, 2019.

[9] S. R. Chhetri, A. Canedo, and M. A. A. Faruque, "Kcad: Kinetic cyber-attack detection method for cyber-physical additive manufacturing systems," in *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2016, pp. 1–8.

[10] T. J. Coogan and D. O. Kazmer, "Bond and part strength in fused deposition modeling," *Rapid Prototyping Journal*, vol. 23, no. 2, pp. 414–422, 2017.

[11] M. A. A. Faruque, S. R. Chhetri, S. Faezi, and A. Canedo, "Forensics of thermal side-channel in additive manufacturing systems," in *CECS Technical Report No.16-01*, 2016.

[12] Y. Gao, B. Li, W. Wang, W. Xu, C. Zhou, and Z. Jin, "Watching and safeguarding your 3d printer: Online process monitoring against cyber-physical attacks," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 2, no. 3, pp. 108:1–108:27, Sep. 2018. [Online]. Available: http://doi.acm.org/10.1145/3264918

[13] J. Gatlin, S. Belikovetsky, S. B. Moore, Y. Solewicz, Y. Elovici, and M. Yampolskiy, "Detecting sabotage attacks in additive manufacturing using actuator power signatures," *IEEE Access*, pp. 1–1, 2019.

[14] Y. Han, S. Etigowni, H. Liu, S. Zonouz, and A. Petropulu, "Watch me, but don't touch me! contactless control flow monitoring via electromagnetic emanations," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: ACM, 2017, pp. 1095–1108. [Online]. Available: http://doi.acm.org/10.1145/3133956.3134081

[15] A. Hojjati, A. Adhikari, K. Struckmann, E. Chou, T. N. Tho Nguyen, K. Madan, M. S. Winslett, C. A. Gunter, and W. P. King, "Leave your phone at the door: Side channels that reveal factory floor secrets," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: ACM, 2016, pp. 883–894. [Online]. Available: http://doi.acm.org/10.1145/2976749.2978323

[16] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 4, pp. 320–327, 1976.

[17] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security)*. Berlin, Heidelberg: Springer-Verlag, 2007.

[18] S. B. Moore, J. Gatlin, S. Belikovetsky, M. Yampolskiy, W. E. King, and Y. Elovici, "Power consumption-based detection of sabotage attacks in additive manufacturing," *CoRR*, vol. abs/1709.01822, 2017. [Online]. Available: http://arxiv.org/abs/1709.01822

[19] M. M. Moya and D. R. Hush, "Network constraints and multi-objective optimization for one-class classification," *Neural Networks*, vol. 9, no. 3, pp. 463 – 474, 1996. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0893608095001204

[20] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443 – 453, 1970. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0022283670900574

[21] I. Oregi, A. Pérez, J. Del Ser, and J. A. Lozano, "On-line dynamic time warping for streaming time series," in *Machine Learning and Knowledge Discovery in Databases*, M. Ceci, J. Hollmén, L. Todorovski, C. Vens, and S. Džeroski, Eds. Cham: Springer International Publishing, 2017, pp. 591–605.

[22] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, February 1978.

[23] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intell. Data Anal.*, vol. 11, no. 5, p. 561–580, Oct. 2007.

[24] C. Song, F. Lin, Z. Ba, K. Ren, C. Zhou, and W. Xu, "My smartphone knows what you print: Exploring smartphone-based side-channel attacks against 3d printers," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: ACM, 2016, pp. 895–907. [Online]. Available: http://doi.acm.org/10.1145/2976749.2978300

[25] L. D. Sturm, C. B. Williams, J. A. Camelio, J. White, and R. Parker, "Cyber-physical vulnerabilities in additive manufacturing systems: A case study attack on the .stl file with human subjects," *Journal of Manufacturing Systems*, vol. 44, no. Part 1, pp. 154 – 164, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0278612517300961

[26] A. Wang, "An industrial strength audio search algorithm," in *4th International Conference on Music Information Retrieval*, Baltimore, MD, Oct 2003.

[27] M. Wu, Z. Song, and Y. B. Moon, "Detecting cyber-physical attacks in cybermanufacturing systems with machine learning methods," *Journal of Intelligent Manufacturing*, vol. 30, no. 3, pp. 1111–1123, Mar 2019. [Online]. Available: https://doi.org/10.1007/s10845-017-1315-5