A Physical Hash for Preventing and Detecting Cyber-Physical Attacks in Additive Manufacturing Systems

Joshua Erich Brandman

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science
in
Mechanical Engineering

Christopher B. Williams
Jaime A. Camelio
Xiaoyu R. Zheng

May 2, 2017
Blacksburg, VA

Keywords: additive manufacturing, 3D printing, cyber-physical security, physical hash, *in situ* monitoring, side-channel measurement

A Physical Hash for Preventing and Detecting
Cyber-Physical Attacks in Additive Manufacturing Systems

Joshua Erich Brandman

Academic Abstract

This thesis proposes a new method for detecting malicious cyber-physical attacks on additive manufacturing (AM) systems. The method makes use of a *physical hash*, which links digital data to the manufactured part via a disconnected side-channel measurement system. The disconnection ensures that if the network and/or AM system become compromised, the manufacturer can still rely on the measurement system for attack detection. The physical hash takes the form of a QR code that contains a hash string of the nominal process parameters and toolpath. It is manufactured alongside the original geometry for the measurement system to scan and compare to the readings from its sensor suite. By taking measurements *in situ*, the measurement system can detect in real-time if the part being manufactured matches the designer's specification. A proof-of-concept validation was realized on a material extrusion machine. The implementation was successful and demonstrated the ability of this method to detect the existence (and absence) of malicious attacks on both process parameters and the toolpath.

A case study for detecting changes to the toolpath is also presented, which uses a simple measurement of how long each layer takes to build. Given benchmark readings from a 30x30 mm square layer created on a material extrusion system, several modifications were able to be detected. The machine's repeatability and measurement technique's accuracy resulted in the detection of a 1 mm internal void, a 2 mm scaling attack, and a 1 mm skewing attack. Additionally, for a short to moderate length build of an impeller model, it was possible to detect a 0.25 mm change in the fin base thickness.

A second case study is also presented wherein dogbone tensile test coupons were manufactured on a material extrusion system at different extrusion temperatures. This process parameter is an example of a setting that can be maliciously modified and have an effect on the final part strength without the operator's knowledge. The performance characteristics (Young's modulus and maximum stress) were determined to be statistically different at different extrusion temperatures (235 and 270 °C).

A Physical Hash for Preventing and Detecting
Cyber-Physical Attacks in Additive Manufacturing Systems

Joshua Erich Brandman

General Audience Abstract

Additive Manufacturing (AM, also known as 3D printing) machines are cyber-physical systems and are therefore vulnerable to malicious attacks that can cause physical damage to the parts being manufactured or even to the machine itself. This thesis proposes a new method for detecting that an AM system has been hacked. Attacks are identified via a series of measurements taken by a measurement system that is disconnected from the main network. The disconnection ensures that if the network and/or AM system are hacked, the manufacturer can still rely on the measurement system for attack detection. The proposed method uses a *physical hash* to transfer information to the disconnected measurement system. This physical hash takes the form of a QR code and stores in it the nominal process parameters and toolpath of the build. It is manufactured alongside the original geometry for the measurement system to scan and compare to the readings from its sensor suite. By taking measurements in real-time, the measurement system can detect if the part being manufactured matches the designer's specification. A proof-of-concept of the proposed method was realized on a common AM system. The implementation was successful and demonstrated the ability of this method to detect the existence of a malicious attack.

A case study for detecting changes to the toolpath is also proposed using the simple measurement of how long each layer takes to build. Given benchmark readings of a part manufactured on the same technology as the proof-of-concept implementation, several modifications were able to be detected. The attack types tested were the insertion of an internal void, scaling the part, and skewing the part. A second case study is also presented where components were manufactured at different extrusion temperatures. By measuring the force required to break the parts, it was determined that temperature has an effect on the final part strength. This confirmed that malicious attacks targeting extrusion temperature are a plausible threat, and that the parameter should be measured in the proposed system.

Acknowledgements

I would first like to thank my parents for their endless support and for listening to me whine every time I had an idea and discovered a reason it would not work. I would like to thank Dr. Chris Williams, my advisor, for guiding me through this challenging and stressful year. I am grateful for the suggestions provided by Dr. Jules White in the development of this method. I would also like thank Logan Sturm for his help and contributions to this work, Cam Chatham for helping me run SLS prints, Callie Zawaski for allowing me to use her modified DreamVendor printer, and Robert Mills for his help with tensile tests. I also thank the entire DREAMS Lab for moral support.

Table of Contents

Table of Figures

Chapter 1 – Introduction

Additive manufacturing (AM) has become an increasingly common method of manufacturing both functional prototypes and end-use parts. The material selection and the robustness of these machines have seen continual improvement, to the point that numerous companies have started to use AM to manufacture components that would otherwise be much more expensive and take longer to manufacture with traditional methods [1], [2]. Notable examples of AM in industry include SpaceX's SuperDraco thrusters [3], near-end-of-life component repairs on military helicopters with LENS systems [4], [5], and General Electric's LEAP jet engine nozzles and ultra-lightweight brackets [6], [7]. Unfortunately, malicious attackers have taken note of AM processes as an attack surface, and have become a serious threat to these companies and their adoption of AM.

1.1 Cyber-Physical Systems in History

The modern manufacturing world is in the process of shifting toward 'Industry 4.0,' where all physical machinery is connected via networks and software – an Internet of Things [8]. This departure from standalone equipment is marked by a rise of the cyber-physical system (CPS). A CPS is defined as any machinery that connects the software and hardware domains. The number of CPSs in industry is increasing, and their applications are seen everywhere from aerospace to infrastructure to AM [9]. These different types of CPSs can range in complexity and influence, but many control machinery that is vital to a company's bottom line, or in some cases, whole cities [10]. As the importance of CPSs has increased, so has the likelihood of an attack on them from malicious entities. These attacks can come from a variety of sources, ranging from hackers trying to damage a company, to governments trying to gain an advantage over enemy countries [11], [12].

One of the most well-known examples of an attack on a CPS is the Stuxnet worm that infected a uranium enrichment lab in Iran. The reason that this particular attack has gained so much notoriety since its discovery in 2010 is that it is one of the first documented examples of targeted "malware that had destructive effects in the physical world" [13]–[15]. It set a precedent for countless other CPS attacks and changed the dynamic of cyber warfare. The worm's function was to speed up and slow down the lab's centrifuges past their normal operating points, breaking them much more quickly than they would under normal wear, thus weakening Iran's nuclear capabilities [15]. The attackers' goal was not to draw attention to the fact that they were destroying these machines (which could have been considered an act of war), but rather to do so discretely in a manner that no one would even expect was an attack [11], [15].

The operators of this facility had ensured that their computer network was disconnected from the open internet, likely to prevent this kind of scenario. However, Stuxnet still achieved its goal of taking almost one thousand centrifuges out of operation [16]. The worm entered the lab on a USB stick carried in by one of the workers. It is possible that the worker was a double agent, but researchers of the incident suspect that it was simply someone who found the thumb drive lying around and was uninformed of the "basic cyber hygiene" of not putting unknown devices into computers on classified networks [13], [15]. This is the second reason that Stuxnet has become so well-known – it is a classic application of social engineering that had consequences in the physical world on a geopolitical level, showing that even if a network is isolated, the machines on it can still be compromised [15], [17].

Even before Stuxnet was the Trojan horse that infected the Trans-Siberian gas pipeline during the Cold War [18]. In 1982, the CIA covertly modified the firmware on equipment being sold to the Soviet Union to contain Trojans, in one case giving the US access to a multitude of

controls and settings for one of the USSR's main gas lines [14], [18], [19]. After waiting for the operators to become familiar with the equipment, the altered firmware "reset pump speeds and valve settings to produce pressures far beyond those acceptable to the pipeline joints and welds," resulting in an explosion equivalent to a three kiloton nuclear weapon being detonated [18]. Just as in the case of Stuxnet, there was no direct connection between the pipeline controls and the outside world. Instead, the malicious attackers were able to exploit a CPS vulnerability and "bridge the air gap" to a system believed to be secure [15].

Another commonality between Stuxnet and the gas pipeline attack is that the infected machines in both were Supervisory Control And Data Acquisition (SCADA) systems [14]. They are CPSs and are mainly used to control and monitor large industrial operations such as power plants, water networks, and even weapons systems [10], [14], [19]. What these two attacks (and now many more) have shown, is that SCADA systems are not only possible to hack, but can cause significant damage in the real world due to their integration with physical machinery and manufacturing equipment.

1.2 Cyber-Physical Systems in Manufacturing and Industry

More recently than Stuxnet, CPSs have become incredibly common in the manufacturing industry, almost by necessity. As processes become more sophisticated, more sensors and real-time process control are required to optimize production lines and maximize throughput. The authors of [19] point out that "controllers are computers," meaning that with the addition of more code to run more advanced factories, the possibility of implementation bugs grows as well, giving hackers more opportunities to gain access.

One increasingly common method attackers use to gain access to CPSs is spear phishing attacks, which usually take the form of a seemingly legitimate email message with a malicious

attachment capable of causing havoc in a company's network [20], [21]. In the database of cyber-physical attacks analyzed in [9] and the short list specific to SCADA systems in [14], recorded hacks came from a plethora of unique entry points. These include "direct physical access…, dial-up connections, VPNs, telco networks, wireless systems and 3rd party connections." The number and variety of these distinct attack vectors reveal that simply cutting off a company's network from the internet does not inherently make its manufacturing equipment secure from outside interference. The combination of increasing motivation to perform attacks and growing opportunities has resulted in more and more attacks being reported each year [9], [14].

Of particular concern is the effect that cyber-physical attacks can have on machined parts, due to their use in almost every facet of the manufacturing industry. A case study performed at Virginia Tech and Vanderbilt University used human subjects to demonstrate the ease with which attacks can be made and go unnoticed [22], [23]. In the experiment, groups of students were tasked with using CAM software to CNC mill a dogbone tensile test specimen. However, the machine code created by the students was not sent to the mill; rather, a virus on the computer altered values in the toolpath to reduce the cross-sectional area of the machined part. In almost half of the groups tested, the attack went undetected.

Work has also been done at Virginia Tech to try to characterize the security of cyber-physical manufacturing systems such as the one just discussed [24]. The goal of this work was to provide manufacturers with an intuition as to how vulnerable their CPSs are to an attack. The researchers developed a "stoplight scale" to illustrate either a low, medium, or high level of vulnerability.

1.3 Additive Manufacturing Vulnerabilities as Cyber-Physical Systems

The focus of this research is cyber-physical attacks on AM systems. AM falls into a specific category of CPS where components are manufactured by gradually stacking 2D layers of discrete height on top of each other. This is contrasted with subtractive machinery such as lathes and mills (also CPSs) that remove material from a solid block until the final form is achieved. As part of the CPS domain, two factors have to be considered when discussing security of AM systems [12]. First is the potential for a malicious entity to gain access to a system with the intention of interfering with its operation or functionality. The second concern is a matter of intellectual property (IP), where an attacker with access to a machine could extract confidential part geometry, toolpath, or process parameters.

Within the first category of attack (malicious modification), one failure mode is the destruction of the machine itself, which could waste a company's time and money. This was accidentally demonstrated in 2013 when an AM system manufactured by Powderpart Inc. exploded when it "failed to contain known sources of potential ignition, such as titanium and aluminum alloys" [25], [26]. This particular incident was deemed unintentional, but a hacker with access to an AM system with fine powders (such as this one) could potentially alter the "configuration settings…to allow the device to overheat," resulting in a similar effect or at the very least rendering the machine unusable [25], [26].

Another failure mode in the first category is modification of the process parameters or part geometry. This is of particular concern to companies using AM to create functional prototypes or production parts. Many of these end-use components can see significant stress and fatigue during their life. If the process parameters or toolpath is maliciously modified without the operator's knowledge, this could have destructive effects when the component is put into use and breaks before the designer intended. Geometry modifications in particular have been explored in depth

by researchers at Virginia Tech [27]. Additionally, the authors give examples of attacks that can be made on STLs: insertion of internal voids, scaling the part, indentations/protrusions, and moving individual vertices. Voids, in particular due to their nature of being enclosed inside the part, can be very difficult to detect with conventional measurements or by visual inspection in post. To illustrate the destructive effects of undetected modifications, the authors embedded small voids inside dogbone tensile coupons and noted that they broke significantly before the unaltered controls.

The concept of an STL attack was expanded to a real-world demonstration in [28] where the authors precisely controlled when an AM drone propeller would break by discreetly embedding defects in a CAD file without the operator's knowledge. In a spear phishing attack, they hacked into the victim's computer and replaced the 'good' file with an altered version. After manufacturing the modified propeller, the victim assembled the drone, flew it, and watched the multirotor fall from the sky when the propeller broke mid-flight. Despite the contrived nature of the hack, this attack vector is very plausible and plainly demonstrates that there is a need to monitor that an AM system is creating the expected toolpath with the expected process parameters.

The second attack category that industries and governments are fearful of is theft of confidential IP. Certainly if an attacker gains control of an AM system, they could have access to the toolpath and parameters of the build. More challenging but definitely doable as evidenced by research at UC Irvine, University at Buffalo, and others, is reconstruction of the original 3D geometry [29]–[31]. Part of the reason that companies are so protective of their part-dependent, expertly tuned parameters is simply the cost of tuning. Particularly for AM of metals, which is common for high-strength end-use parts, many iterations on the parameter set may have to be done before a component can be considered optimal [32]. This balance of part shape, orientation, and

process parameters requires a very specific skillset, so companies that have this capability try very hard to protect their competitive advantage. Due mainly to the high price tag on industrial-scale AM machines, many companies choose to outsource their designs to specialized shops [7]. Companies like this are particularly vulnerable to cyber-physical attacks due to the fact that their outsourcing requires trust of the third-party manufacturer. This use-case motivates the research presented in this thesis. A method is needed for companies to securely contract out manufacturing without their IP being stolen and without their components being altered by malicious attackers.

1.4 Disconnected Side-Channel Measurement System

With an increasing number of means and motivations for CPSs, particularly AM machines, to be compromised, research into how best to protect CPSs has grown as well. As noted earlier, it is difficult to guard every entry point an attacker might take to access a system, be it a remote wireless connection, physical access, or some other method. The focus of this thesis is not to protect the entry points of CPSs. Rather, emphasis is placed on detecting that an attack has occurred and informing the operator.

The failure modes of AM parts due to malicious attacks can be grouped into changes to the toolpath and changes to the process parameters. Because of the capability of AM to produce parts with complex and spatially varying macro- and microstructures, traditional measurement techniques such as CMMs, structured light, and even CT scanning are not always sufficient to fully characterize a part after it has been manufactured [33]. If a modification (such as the introduction of an internal void or a lowered extrusion temperature) is made to the part in an area that would be covered up by subsequent layers, traditional measurement techniques in post would not necessarily be able to detect the error. To prevent this scenario from occurring, many companies ensure quality

control by performing *in situ* process monitoring on their AM systems to detect these errors as they occur.

For the purposes of this thesis, a side-channel is defined as a method in which information can indirectly be obtained about a process. This work takes advantage of side-channel measurements by placing a suite of sensors inside of an AM machine to take *in situ* readings throughout the build. The purpose of these measurements is to account for the very situation described above where an attacker has gained access to an AM system and is attempting to make malicious changes. For this detection method to be an effective way of determining if an AM machine has been compromised, the measurement system will have to measure both the relevant process parameters as well as the geometry of each layer (i.e., toolpath). Additionally, the comparison between what the sensors see and the specification provided by the designer will have to be done without exposing information about the IP of the build.

Since this thesis explores the possibility that an AM system's network has been compromised, the base of trust in the proposed detection method becomes the side-channel measurement system itself. In order to avoid any possibility of the measurement system becoming infected, however, it will have to be entirely disconnected from both the network and AM system. This introduces the challenge of communicating to the measurement system the specification against which its readings should be compared.

1.5 Goals for this Research

The overall goal of this research is to protect AM systems from cyber-physical attacks and prevent the theft of valuable IP, all while ensuring quality of the part. In order to accomplish this, the presented methodology centers on a *physical hash* that couples cyber and physical data. The

system is composed of a side-channel measurement system and a hash of physical part characteristics that is represented by an additively manufactured object.

The remainder of this thesis is structured as follows: Chapter 2 will synthesize the prior work in this field and identify the current knowledge research gaps and opportunities. In Chapter 3, the concept of a physical hash is presented. In Chapter 4, the disparate pieces that are needed to use a physical hash in practice are explained and demonstrated. These include transferring information via AM with QR codes, using predefined ranges in the physical hash, a toolpath measurement technique, and the verification of a process parameter's effects on final part strength. Finally, in Chapter 5, a methodology for using a physical hash will be realized in a proof-of-concept implementation as a validation test.

Chapter 2 – Literature Review

Significant research has been done to prevent and detect cyber-attacks, but less has been done for the prevention and detection of cyber-*physical* attacks [34]. There is a key gap in the literature regarding how to prevent and detect attacks on CPSs, and specifically AM systems. The ideas that will be discussed in following chapters rely and build on prior work in an attempt to address this gap.

The first area of prior work that will be borrowed is taxonomies for discussing attacks on CPSs. The next area is *in situ* monitoring as it applies to AM. Other key concepts that will be used in this thesis are QR codes and hash functions as a means of securely transferring information; prior work in these areas will be discussed as well.

2.1 Language and Taxonomy of Cyber-Physical Attacks

Several taxonomies have been proposed as a means to discuss cyber-only attacks, with varying levels of success and adoption by the community. Many of them are too narrow in scope and are limited to a particular aspect of security, such as wireless communication [35], [36]. Others, such as the taxonomies proposed in [37] and [38], have been more successful; the work in [37] has even been adopted by the US Computer Emergency Readiness Team (US-CERT) as their classification system of choice [34]. However, both of these taxonomies still face the limitations of being scoped solely to cyber domains and not generally supporting cross-domain CPS descriptions.

The first step to making CPSs as secure as the cyber-only world is to define a language for discussing them. Work at Vanderbilt University and Virginia Tech has been done to define taxonomies for classifying cyber-physical attacks as well as connecting them with quality inspection measures [39]. NIST has also developed a taxonomy in a similar vein and compiled a vulnerability database with a search engine front-end [40]. Additionally, the authors of [34]

introduced CP-ADL, or the Cyber-Physical Attack Description Language, based on their previous work in [35] defining a six-dimensional taxonomy. These taxonomies are significant as they create a unified and formalized set of descriptors that can fully qualify a cyber-physical attack, making it easier to analyze any given attack as well as compare it with others. However, there is still a gap in the literature regarding how to actually combat cyber-physical attacks.

2.2 Hash Functions

Cryptographic hash functions have been around since the 1970s and are 'one-way' algorithms that take an arbitrary amount of data as input and return a fixed amount of data as output [41]. The output, or digest, can be represented as a string with as few as 32 characters in some cases. This small size allows hash strings to be represented efficiently in a moderately sized QR code. The advantages of this property will be seen in the subsequent chapters and explored in more depth in Section 4.1. Hash functions are considered one-way because there is no inverse function to return the preimage, or input, given the message digest. Note that this is distinct from encryption, where a decryption function does exist. Another property that hash functions exhibit is the 'waterfall effect.' Any change to the input, no matter how slight, results in a completely changed output.

2.3 *In Situ* Measurements

As mentioned earlier, AM parts can fail due to attacks on the process parameters or on the toolpath. To ensure that these attacks do not go unnoticed, a side-channel measurement system can be used. Significant research effort is currently underway in creating *in situ* measurement systems for AM systems for the purposes of ensuring final part quality. The current literature can be divided into the same categories as the attack types: process parameter modifications and part geometry/toolpath modifications.

2.3.1 Process Parameters

AM part properties are sensitive to changes in process parameters. These sensitivities serve as the primary motivation for research in *in situ* measurement systems. The dissertation on powder bed fusion (PBF) in [42] points out a number of key parameters in determining the quality of a part, including layer thickness, hatch spacing (distance between lines the laser traces), scan speed, and laser power. None of these parameters would be easy to measure by eye, but all affect important metrics such as final density, surface finish, and especially mechanical strength [42]. The same holds true for material extrusion. The authors of [43] and [44] researched the effects of various process parameters on the mechanical properties of AM parts, and found that extrusion rate, nozzle and bed temperatures, linear nozzle speed, and layer height all had significant impacts. These papers emphasize the motivation to have an instrumented side-channel measurement system to read and compare these parameters to the operator's specification in real-time.

With this motivation, a fair amount of research has been done looking into the technicalities of how *in situ* measurements of process parameters should be made, mostly with an emphasis on quality control. A specific type of PBF, Selective Laser Melting (SLM), is another AM process that is commonly used for end-use parts. The authors of [45] provide an extensive survey of methods that have been studied for measuring the over 50 process parameters in SLM systems. Their discussion includes methods such as electromagnetic meltpool signature monitoring, Lagrangian and Eulerian reference frame modalities, and imaging with optical and thermal cameras. Many of these technologies are even already in use in commercial monitoring systems [46]–[48]. For an all-inclusive *in situ* measurement system that can capture all of the relevant process parameters, it is likely that all of these technologies would have to be employed in some form.

There is one aspect that none of these systems seem to take into account: security. All of the sensor suites in the above papers are by necessity integrated with the AM system and are therefore as vulnerable to an attack as the AM system itself. They all focus on quality assurance and do not address the question of how to prevent an attacker from targeting the measurement system. If the measurement system is compromised, an attacker would be able to simply assert that the measured value was within specification. This motivates why the *in situ* measurement system must be offline in order to serve as a base of trust.

2.3.2 Geometry and Toolpath

Existing measurement systems that monitor the manufactured geometry and toolpath face the same challenges as those looking at process parameters. Some commercial systems, such as QMmeltpool, are able to reconstruct the manufactured part in digital 3D based on the tomography of each layer, but this is specifically designed for SLM systems and is incredibly expensive [47]. Other techniques that can apply to multiple additive technologies are presented in [49]. For example, the article suggests the use of accelerometers embedded in a material extrusion nozzle to detect "potential anomalies" based on vibrations, or ultrasound sensors to "ensure the final part is free of internal voids" [49]. Another example of an *in situ* measurement technique it puts forward is high-resolution video of the powder bed. In [50], the researchers used a camera filming at over 16k frames per second (fps) to capture data about the meltpool in a PBF system. They were able to get accurate measurements down to 10 μm per pixel, but at the cost of producing 2.3 TB of data per hour of build time. In [33] and [51], the researchers bonded a piezoelectric transducer to a material jetted part to measure its piezoelectric response over a frequency sweep. If a change was made to the part, such as a slight increase in mass or the insertion of an internal void, a measureable change could be found between the produced waveforms.

There has been a significant amount of promising work in the field of toolpath measurement, but none of the research has addressed how to leverage the techniques to provide security for AM systems. Indeed, for many of the methods that have been proposed, they themselves provide another attack surface if left connected to the AM system and network. This is the same gap in knowledge that was seen for quantifying process parameters.

2.4 Information Transfer via AM Systems

One method of ensuring the security of an AM component and its process parameters is to embed extra information in the build about what the measurement system should see. By transferring information via the AM system and not over Ethernet or wirelessly, the measurement system can be 'air-gapped' and thus protected from malicious attackers more effectively. This will be discussed in greater detail in Chapter 3.

One example of embedding information in an AM part is shown in [52], where the author demonstrates a method to create barcodes with vat photopolymerization using only a single material. At the same time, high contrast is kept between different logical pixel values. By embedding a code like this into a part, it provides a machine-readable way to "track parts through manufacturing and distribution" [52]. The information contained by the barcode could be a simple part identifier as the author demonstrates, used to match manufactured components with CAD data, or it could even be a list of the process parameters needed to manufacture itself, as will be explored in this thesis.

Another slightly different example is shown in [53], where the authors embedded information in a 3D barcode (essentially several 2D barcodes stacked on top of each other). They then used a terahertz imaging system to scan the code's volume in 3D and retrieve the information encoded in it. In their paper, only the outer case of the barcode is made with AM, not the individual

layers. However, the idea is the same as in [52], that when paired with another physical item, these small barcodes can be used for object tagging, inventory management, and even a small amount of data storage.

The author proposes the concept of a physical hash for protecting AM systems, which draws on prior work in the areas of hashing, *in situ* monitoring of AM systems, and QR codes. These fields will be combined to create a methodology capable of securely detecting the integrity of an AM part's process parameters and toolpath while keeping its IP hidden.

Chapter 3 – The Physical Hash and System Overview

For reasons of security and the protection of IP, the concept has been introduced of using *in situ* side-channel measurements as a means of detecting unwanted changes in an AM system. The physical hash then takes the role of linking the cyber data of the part to the physical world and the disconnected measurement system. This raises the question of how to securely transfer the information of the physical hash across this 'air-gap.'

3.1 The Physical Hash

The physical hash is a novel concept defined as a secure representation of the process parameters and toolpath of the designer's part. It is used to transfer information to the 'air-gapped' measurement system about nominal values and allowable tolerances of the build's process and toolpath parameters. This is done without exposing this valuable IP to an attacker or even to the manufacturer. The physical hash takes the form of a QR code and stores a single hash string of the physical part characteristics. The QR code is converted to a 3D STL and is manufactured alongside the original part. The measurement system uses its sensor suite to perform side-channel measurements during the build and scans the QR code to compare its readings to specification. A schematic of this concept is shown in Figure 3.1.

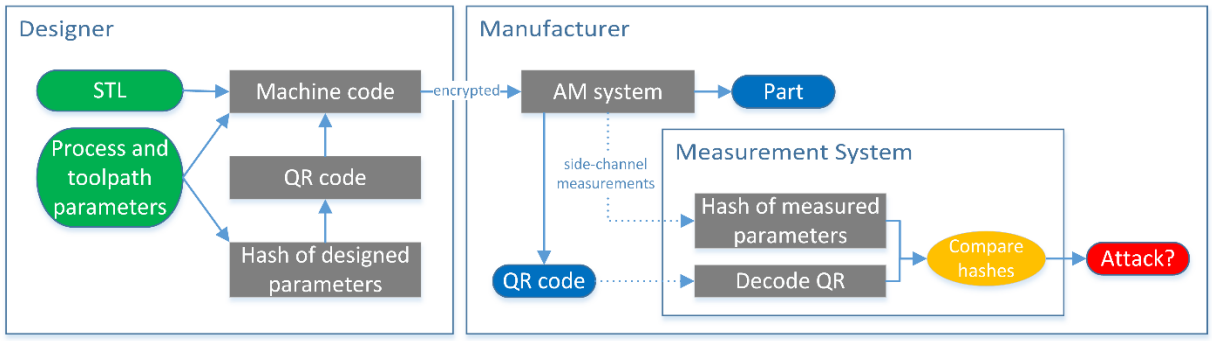

Figure 3.1. High-level schematic of the proposed system architecture.

3.2 Rationale for the Physical Hash's Manufacturability, Format, and Contents

Three elements of the physical hash must be addressed in order to define its use. First, the question of how the physical hash will be manufactured is considered. Next, the format with which it is represented is determined, and finally, the question of how the physical hash will store information is explored.

3.2.1 Manufacturability of the Physical Hash

The first question to consider is if the physical hash even needs to be manufactured. Why not have the manufacturer manually enter the values into the measurement system by hand? The first argument against this is the requirement of IP protection. The designer does not want their carefully constructed process parameters and toolpath information to be known to anyone but the AM system itself. Another argument for manufacturing the physical hash is simple logistics. It is possible to enter a few values by hand, but many industrial AM systems have upwards of 50 variables or even more, most of which can have direct and meaningful impacts on the final part quality [45]. This very quickly becomes infeasible and opens the door to human error or even intentional malicious modification.

Since the physical hash does indeed have to be manufactured, the next question is if it is necessary to create it on the same system used to manufacture the original part. For example, one might consider manufacturing the QR code on a normal 2D office printer for the reason that it would require significantly less time and cost. However, this is explicitly not a valid alternative. By introducing an additional machine into the system architecture, attackers are provided with a multitude of new attack vectors. The function of the physical hash is to link and integrate the digital design data with the part. By splitting their manufacture to separate machines, the security associated with this integration is inherently lost.

3.2.2 Format of the Physical Hash

The second element of this method to consider is the format in which the physical hash will be stored. The major requirement is that the information has be interpreted by the measurement system's sensor suite and cannot be transferred by normal networking methods. The chosen solution is the QR code and leads to the first research question (see Section 3.7). QR codes have been established as a reliable way to transfer information via optical image processing techniques. They lend themselves to AM due to the fact that many AM processes involve a contrast change as each layer is created. This makes it easy for a camera to read and interpret the QR code in a fast-paced production environment. This concept was introduced in Section 2.4 and will be explored in more detail for multiple AM processes in Section 4.1. Additionally, compared to other barcode-style formats, QR codes can store as much as ten times the amount of data as a comparably sized 1-dimensional barcode [54]. However, they do have practical size limitations that will motivate the discussion in the next section.

3.2.3 Contents of the Physical Hash

The third matter to be addressed is the question of what information the QR code needs to store. In order for the measurement system to compare its readings to the specification, the physical hash will need to be able to transfer the nominal values for each parameter as well as the corresponding tolerances.

First consider storing this information as plaintext in the QR code. A single, delimited string contains the parameters' nominals and tolerances. While this is the simplest solution, it is lacking in two key areas. First, if an attacker does have access to the AM system and its toolpath, they will be able to decode the QR code and gain access to the IP it contains. Additionally, depending on the number of parameters to be measured, there is a practical limit of how much information can reasonably be stored in a QR code.

To deal with the problem of IP protection, one might consider encrypting the string. This does solve the issue of unprotected IP, but there is still a physical limit that prevents storing large amounts of data in a small barcode footprint. Even before considering the size of the message itself, when signed and encrypted with PGP for a 2048-bit key, a single-character string becomes a text file of 4472 bits (over half a KB). The smallest possible QR code that could store this is an 81x81 grid [55]. To be adopted by industry, the physical hash must be relatively small and not use a significant amount of material or build space.

A solution that solves both the IP and space problems is hashing the string. Recall from Section 2.2 that hash functions are one-way algorithms that convert any amount of data to a fixed length string. The fact that the output is of fixed length means that no matter how many parameters the measurement system needs to check, they can all be transferred via a single QR code. Additionally, if an attacker gains access to the QR code, they will not be able to preimage the IP from the hash string. This also means, however, that the measurement system cannot see the information that went into creating the hash, and thus cannot directly compare its measurements to the nominals.

The reason that direct comparison with the hash string is not possible is that by definition, hash functions do not allow for a tolerance. This is due to the waterfall effect, illustrated in Figure 3.2 with an example of a characteristic process parameter (laser power) and nominal value. Note that if the input is even slightly changed, the output is completely different. The key limitation is that physical measurements are not perfect. If any measured value differs from the nominal at all, direct comparison of the hash strings is useless as a verification of system integrity.

```
MD5("laser_power=10.0") → 0ab23e8a5ef024d29c7572bf028ac381
MD5("laser_power=10.1") → 7f6d9bf11fd7d5e30abc37af75803a12
```

Figure 3.2. Demonstration of the waterfall effect using the MD5 hash function. Small changes to the input result in drastic changes to the output.

3.3 Hashing with Predefined Ranges

Here, the idea is introduced of hashing not a set of nominals and tolerances, but the ranges or 'bins' in which the nominals fall. As explained above, hashing exact values does not allow for any tolerance in measurements. Imagine instead, that both the program creating the physical hash as well as the measurement system are aware of multiple sets of predefined ranges. These ranges are defined at several tolerance levels for every measureable parameter. By defining the information stored in the physical hash in this manner, any parameter that can be quantized into a range (or multiple ranges) can be securely transferred to the measurement system. This motivates the second research question (see Section 3.7) of how predefined ranges can be used in the physical hash.

This concept is best explained with an example. Consider the same process parameter used in Figure 3.2, laser power, with a nominal value of 10 W and acceptable tolerance of ±0.5 W. When the measurement system is initially installed, it is loaded with several sets of possible laser power ranges. The program that the designer uses to create the QR code has the same sets of ranges. The designer tells the program to choose the set of ranges corresponding to the acceptable tolerance, and the following set is loaded into memory: `...,[8.5-9.5),[9.5-10.5),[10.5-11.5),...`. The designer tells the program that the nominal value is 10 W, and the corresponding range `[9.5-10.5)` is automatically chosen. The string stored in the QR code, then, is not the hash of "`laser_power=10`", but the hash of "`laser_power=[9.5-10.5)`".

During operation of the AM system, the measurement system reads a laser power of 10.1 W. At no point is it aware that the nominal value is 10 W. It does, however, see that 10.1 falls in the range `[9.5-10.5)`, and hashes "`laser_power=[9.5-10.5)`". The measurement system scans the QR code with a camera and sees that the hash strings match, indicating normal

operation. Had the measurement been 10.6 W instead, the chosen range would have been `[10.5-11.5)`, the resulting hash would have been different, and the strings would not have matched, indicating the presence of an attack.

By careful consideration of the AM process in which the measurement system is installed, 'tolerance presets' can be created to allow for different use-cases. These presets let the designer mix and match ranges of differing width for each of the parameters being monitored. This gives a flexibility to the system that mitigates the limitation of having to predefine ranges. The other advantage of having presets is that the only manual step required of the manufacturer (other than starting the build) is telling the measurement system which preset to use.

3.4 Detailed Steps for the Creation and Use of the Physical Hash

The proposed method shown at a high level in Figure 3.1 can be decomposed into the following detailed steps:

1. *Predefined ranges established*. At the time of installation, the measurement system is loaded with all sets of ranges for every parameter and tolerance level (i.e., range width).

2. *Designer defines STL/process parameters/toolpath parameters*. After the designer has created the STL to be manufactured, they define the process parameters of the build and allowable tolerances. Toolpath parameters may be defined based on preexisting benchmark components or from the file itself. Specified tolerances describe a 'preset' by which the operator can later tell the measurement system which range sets to choose.

3. *Designer creates QR code physical hash*. After the designer enters the nominal process and toolpath parameters, the following sub-steps happen automatically.
    i. A set of ranges is chosen based on the tolerance preset specified by the designer.
    ii. Corresponding ranges are chosen for each parameter and concatenated into a string.
    iii. The string is sent through a hash function.
    iv. The resulting hash string is encoded in a QR code and converted to an STL

4. *Designer exports machine code*. The designer loads the original STL, the STL of the physical hash, and the process parameters into the appropriate toolpath generation software. The resulting machine code is then exported.

5. *Designer sends machine code to manufacturer*. Using the company's preferred method (email, peer-to-peer, etc.), the machine code file and tolerance level preset are encrypted and sent to the manufacturer.

6. *Manufacturer begins build*. Based on the information from the designer, the manufacturer tells the measurement system which preset to use for the ranges and loads the machine code into the AM system.

7. *Measurement system performs in situ side-channel measurements*. Throughout the build, the measurement system uses its sensor suite to take readings.

8. *Measurement system compares readings to physical hash*. The measurement system scans the QR code and decodes it to obtain the physical hash string. Depending on the designer's requirements, the comparison to the physical hash can occur every layer, or average multiple layers and then compare. The following sub-steps happen automatically.
    i. Corresponding ranges are chosen for each measured parameter and concatenated into a string.
    ii. The string is sent through a hash function.
    iii. The resulting hash string is compared to the string in the QR code. If they do not match, the system has been attacked and the build should be aborted.

3.5 A Possible Failure Point of the Physical Hash

There is likely not a foolproof solution to the problem of cyber-physical security, and one should assume that a determined and sufficiently knowledgeable attacker will always be able to hack into a system. However, the method proposed in this thesis does not claim to stop every attack; rather it increases the investment required of an attacker to a much greater level than the investment of the manufacturer.

A possible, albeit unlikely, failure point of the physical hash occurs if an attacker changes the toolpath of the QR code itself. This is difficult, though, because by manufacturing the physical hash on an AM system, the proposed methodology enforces that not only does the attacker have to gain access to the machine to modify its settings, they also have to simultaneously gain access to its toolpath, figure out how to hash their modified parameters to match the parameters the measurement system records, turn that hash into the 3D STL of a QR code, generate machine code for it that is compatible with the AM system, replace the existing machine code with the attacked

version, and still stay undetected throughout the whole process. In short, while an attack like this is technically possible, a significant amount of effort and cross-disciplinary knowledge is required to coordinate a successful attack.

3.6 Trust Scenarios and Likely Use Cases

The methodology outlined in this section proposes that the designer send the machine code containing the original part and physical hash to the manufacturer. This is seen as the most likely use case of the proposed system. Very little trust of the manufacturer is required and as much of the human element as possible is removed from the process.

Many companies fall in this category, but there are several more that do not have the expertise required to generate the machine code themselves. Instead, they rely on the manufacturer to define the appropriate parameters. The inherent downside of this use case is that the IP of the part geometry has to be released to the manufacturer. Given that a company trusts a manufacturer but not necessarily the integrity of their AM systems, the proposed methodology can be modified to work for this use case. Instead of the designer creating the physical hash and sending the machine code, they can send the part geometry and allow the manufacturer to create the physical hash on their end. Clearly more trust is required of the manufacturer in this case, but if the AM system has been compromised, the physical hash will still be able to detect the modifications.

3.7 Research Questions

Given the above discussion of the proposed method of attack detection, specific research questions can now be summarized. These questions are answered through experimentation on multiple AM systems in Chapter 4. Using the answers to the questions, the complete approach is validated with a case study in Chapter 5.

1. Are QR codes a feasible method to transfer information via AM?
    a. For which AM processes will this idea work?

2. How can predefined ranges and the physical hash be used in conjunction with side-channel measurements to verify the integrity of an AM system?
3. How can the toolpath be measured?
    a. Can the build time for each layer be used to verify toolpath integrity?
    b. What is the smallest attack that can be detected with this method?
        i. For a void insertion attack? scaling attack? skewing attack?
4. What makes an effective process parameter attack?
    a. Is extrusion temperature a necessary process parameter to monitor?

Chapter 4 – Experimentation with Physical Hash Building Blocks

In order to implement the proposed method of verifying the integrity of an AM system, several pieces have to work together. These separate aspects of the overall system are summarized by the research questions in Section 3.7 and will be discussed in detail in this chapter.

4.1 QR Codes as a Means of Transferring Information

The first research question asked if QR codes are a feasible method to transfer information via AM. In this section's experiments, the claim that it is easy to manufacture QR codes with AM and for a camera to subsequently interpret them was tested on both powder bed fusion and material extrusion systems. Using QR code generation software [56], the string "DREAMS Lab" was turned into a QR code. Custom MATLAB code (Appendix A) was written to automate the process of turning the 2D image to a 3D STL. Images of the QR codes manufactured with PBF (Figure 4.1b) and material extrusion (Figure 4.1c) systems are shown below.

Figure 4.1. The a) original QR code and manufactured QR codes on b) powder bed fusion and c) material extrusion systems.

Most standard QR code processing applications were not able to immediately decode the QR codes as they were manufactured. This was mainly due to the fact that the AM systems could not reproduce the code with as much resolution and contrast as is in the perfectly binary image in Figure 4.1a. However, with minimal image processing, the contrast was successfully increased to sufficient levels. For PBF, the contrast came from the saturation of sintered areas of the plastic

powder changing relative to non-sintered areas. For material extrusion, the contrast came from changes in hue. Most modern extrusion systems have at least two extruders for model and support material and should be capable of this. AM processes that were not tested, but should also be able to manufacture readable QR codes include:

- Binder Jetting
    - For similar reasons to PBF, this AM process should also show high contrast due to wetted powder being darker than loose powder.
- Directed Energy Deposition
    - In this AM process, metal powder is directly deposited onto lower layers. Experimentation with this technology is required to determine the feasibility of manufacturing a QR code with sufficient contrast, but with the right settings and image processing techniques, it should be possible.
- Vat Photopolymerization
    - As discussed in Section 2.4, prior work exists manufacturing barcodes with this AM process [52]. The authors were able to differentiate high and low areas by creating a 'crosshatch' pattern on low squares.
- Material Jetting
    - Creating QR codes with this AM process is trivial due to its ability to jet distinct high-contrast materials on a pixel-by-pixel basis.

4.2 Confirmation Experiment on the Use of Predefined Ranges and the Physical Hash

Once the use of QR codes as the vehicle of information transfer was established, the next piece of the overall system that needed to be tested was the concept of using predefined ranges in conjunction with the physical hash. The purpose of this test was to ensure that given a side-channel measurement from a disconnected measurement system, the physical hash could be used to confirm the integrity of that particular parameter. The methodology presented in Chapter 3 was implemented in MATLAB (Appendix A) to allow for (semi)automation of the process and tested on a powder bed fusion system (DTM Sinterstation 2500plus).

4.2.1 Predefined Ranges Confirmation Experiment – Methods

After defining parameters and tolerance levels, the designer runs the appropriate script. The program will first prompt the designer to enter a tolerance preset so that the size of the

allowable ranges can be defined. They will then be prompted to enter the ideal values for each relevant parameter. Behind the scenes, the program first loads the predefined ranges based on the input tolerance preset. It then decides the ranges in which the entered ideal values falls, and concatenates the resulting plaintext.

The script then takes this plaintext and runs it through the MD5 hash function [57]. MD5 was chosen because the length of the resulting string fits into a 29x29 QR code (or sometimes 25x25 depending on the alphanumeric distribution), whereas other hash functions such as SHA-2 or SHA-3, might require a larger 2D representation. MD5 has been shown not to be collision resistant, but it is still secure against a preimage attack [58]. This is important because even if an attacker gains access to the QR code, they will not be able to back out the information it contains. The fact that it is possible to create two strings that map to the same hash is irrelevant for the purposes of the proposed method. The resulting hash is then converted to a QR code [56], which MATLAB then converts to a matrix of ones and zeros. This matrix then gets passed to an OpenSCAD script that converts it into 3D geometry that can be manufactured with AM.

The sensor suite for this simple confirmation experiment was a thermistor mounted inside the AM system measuring the ambient temperature. The appropriate script for the measurement system is executed at the start of the build. The manufacturer enters the same tolerance preset that was used by the designer to generate the physical hash, and then their work is done. The program will first use the preset to load the same set of ranges as determined previously, and then it will begin taking measurements for each layer of the build. After measurements for each parameter have been taken, the script determines the ranges in which the averages fall, concatenates the resulting plaintext, and hashes.

After the QR code has been manufactured, a camera looking at the powder bed takes a picture of it. The picture is then sent through a variety of image processing algorithms to adjust for perspective shift, increase the contrast between high and low areas of the code, and account for the changes in lighting across the part [59]. The result of these techniques is a binary image representing the code, which is then decoded. The string that is returned is then compared to the hash generated from the measurements. If the hashes do not match, it indicates that the part has been manufactured out of specification. Otherwise, no attack has occurred and the integrity of the measured parameter is intact. Since the goal of this experiment was simply to test the concept of a physical hash being used to securely transfer data, the desired result is for the strings to match.

4.2.2 Predefined Ranges Confirmation Experiment – Results

Figure 4.2 shows the dialog that the designer had with the console after running the appropriate script. The parameters that were entered are typical values for the nylon-12 material.

```
>> Enter tolerance preset (1, 2, or 3): 1
>> Enter the nominal ambient temperature (°C): 152
>> Enter the nominal feed temperature (°C): 130
>> Enter the nominal laser power (W): 12
>> Enter the nominal scan speed (m/s): 3
```

Figure 4.2. Dialog for how the designer is asked to input process parameters into the automated program.

The program only asks for four values to be entered as a demonstration of the concept. The tolerance preset defining the widths of the ranges was arbitrarily chosen for this test. For ‘preset 1,’ the temperature ranges have a width of 2 °C, the laser power has a width of 1 W, and the scan speed has a width of 1 m/s. If smaller tolerances are required, however, ‘preset 2’ or ‘preset 3’ could have been chosen. For example, ‘preset 2’ reduces the width of the possible ranges to 1 °C for the temperatures, 0.5 W for the laser power, and 0.5 m/s for the scan speed. The chosen

tolerance levels are purely a function of the machine's controller accuracy and the designer's allowable tolerance. From the entered nominals and tolerance preset:

1. The chosen ranges were concatenated as "`[151,153],[129,131],[11.5,12.5],[2.5,3.5]`".
2. This plaintext was sent through a hash function, yielding the string `6dc720a4651f2b5faab979778c57a87e`.
3. A 2D QR code was created based on the above hash string (Figure 4.3a).
4. A 3D QR code STL was created based on the 2D image (Figure 4.3b).

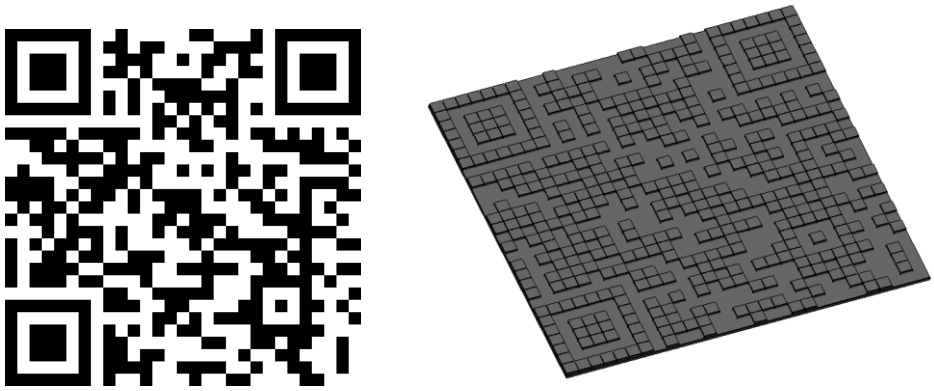

Figure 4.3. a) 2D QR code generated by the MATLAB script and b) 3D STL generated by the OpenSCAD script.

During the build, ambient temperature data was collected with a thermistor mounted inside the AM system. In a more complete implementation, all of the parameters would have had dedicated sensors, but as a demonstration, only the ambient temperature was tracked. The other three parameters were given 'fake' measurements for the purpose of this confirmation test. The measurement system program averaged the ambient temperature readings over the length of the build and found that the average was 151.58 °C, which falls in the range `[151,153]` (Figure 4.4). This was then concatenated with the other 'measured' ranges and hashed.

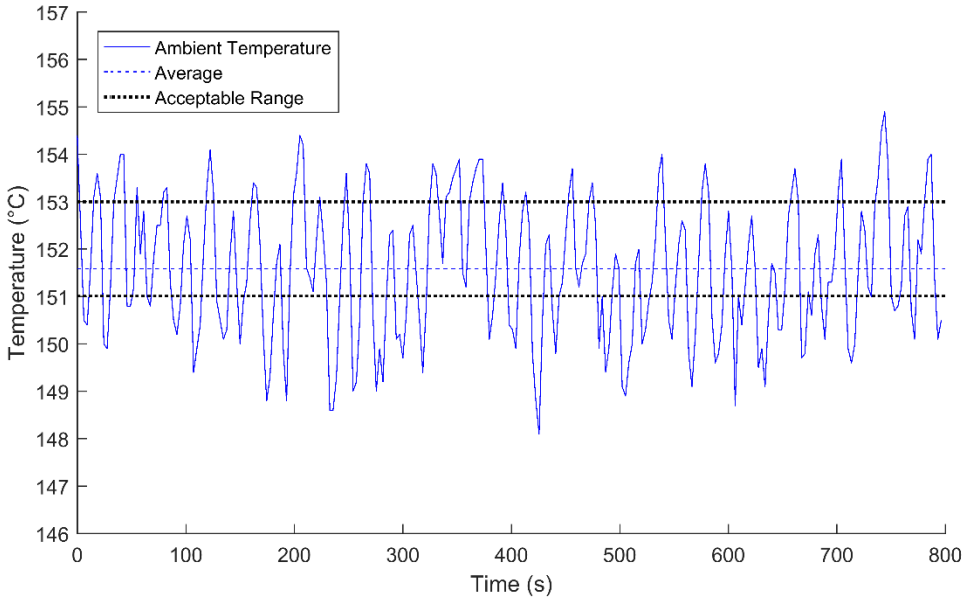

Figure 4.4. Raw ambient temperature data captured over the length of the build.

After the QR code was manufactured, a camera was used to image the powder bed and interpret the code after a small amount of image processing. The string in the manufactured physical hash matched the hash created from the measurement data, which validated that all of the process parameters were within specification.

4.2.3 Predefined Ranges Confirmation Experiment – Discussion

This test demonstrated that the concept of using predefined ranges in conjunction with a physical hash is a feasible way to transfer information to a disconnected measurement system. There are still key pieces that this implementation is missing, however. For example, it has not yet been shown that attacks can be detected. Additionally, this experiment only looked at a single process parameter and has not yet considered toolpath integrity. The next section will look at measuring the toolpath, and all of these concepts will ultimately be incorporated into a single system in Chapter 5.

4.3 Measuring Toolpath with Side-Channel Measurements

One way that AM parts can be attacked is through toolpath modifications. By inserting internal voids or changing key dimensions, an attacker can significantly alter the strength of the final part. There are many ways to measure the toolpath of an AM part, and there is no single solution for every technology. The proof-of-concept implementation of the proposed attack detection methodology will be implemented in Chapter 5 on a material extrusion system. For this reason, this section is concerned with measuring the toolpath on a material extrusion system as well. In an attempt to answer the third research question of how to measure toolpath integrity, a new technique involving the amount of time required to build each layer is presented here. The smallest detectable void insertion, scaling, and skewing attacks using this technique will be established as well.

4.3.1 Time per Layer Approach

As shown in Sturm et al. [27] and discussed in Section 1.3, several types of attacks can be made on a part's toolpath. For the specific attack type of void insertion, the authors state that weighing the part is not sufficient due to the possibility of a void being "filled with a structurally deficient, but equivalently dense material" [27]. Other changes such as scaling or skewing a part may also not be discoverable via mass measurements if the modifications are made in such a way that the total volume is kept constant.

Unlike mass or dimensional changes, one metric that is likely to be affected by these types of attacks is the time required to build affected layers. As such, the author proposes a method of benchmarking the duration spent by the AM system creating each layer to ensure that the toolpath has not been modified in future builds. In general, modern toolpath generation software cannot accurately estimate build times. This means that in order for this technique to be useful, 'good' parts need to have been manufactured and timed so as to acquire a benchmark. Once a benchmark

and reasonable tolerances for the AM process have been established, any part that is manufactured on a potentially compromised system can be timed and compared to the benchmarks.

4.3.2 Experimental Methods for Verification of Layer Time Assessment

The machine used for these tests was an extrusion system, the Solidoodle 3. It was chosen mainly for its ease of access for mounting external sensors (i.e., a GoPro HERO3 camera, see Figure 4.5). Compared to other AM processes, the nozzle on a material extrusion system moves relatively slowly. This made it easier to obtain high resolution time measurements relative to the layer build time. Due to the recent developments of stronger and more durable materials, extrusion systems are becoming more and more common for the manufacture of end-use parts [60]. This is important because it means that this experiment looking at the feasibility of layer times as an attack detection method is immediately applicable to many industries.

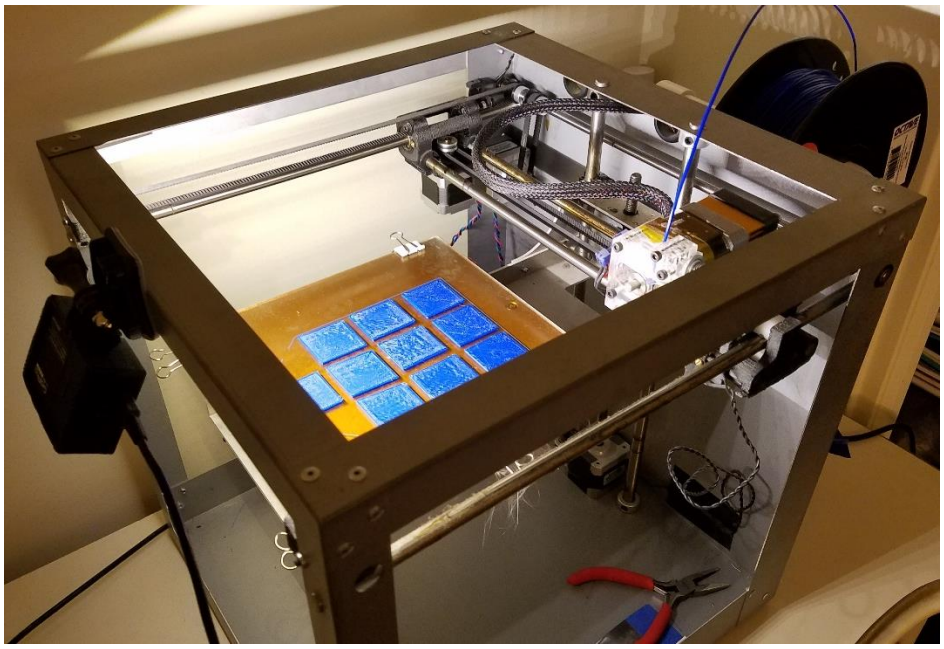

Figure 4.5. Experimental setup for the layer time tests. The camera is mounted to the frame of the machine and pointed at the build tray.

4.3.2.1 Experiment 1 – Individual Layers

Three types of STL attacks were tested using this timing method: a) void insertion, b) scaling the part, and c) skewing the part. For each of the attack vectors, a benchmark was obtained by manufacturing a 30x30 mm square, extruded 1 mm vertically. By mounting a camera on the AM system filming at 30 fps, it was a simple matter to count the number of frames the nozzle spent above each layer and divide by the framerate to obtain a time per layer. For consistency among measurements, the time per layer was defined as the moment when the extruder head starts moving and material starts coming out, to the moment when the extruder leaves the layer to start on the next.

The layer height was set to 0.1 mm, so ten data points were collected (1 [mm] / 0.1 [mm/layer]). A raft layer was also manufactured before the ten layers described above, but it was built more slowly than the rest of the layers to help with bed adhesion, and was thus discarded. In a true implementation of this system, only a single measurement per layer could be obtained since each layer would only be manufactured once. However, for testing feasibility of the method, each of the attack types and levels were also manufactured with ten layers to enable statistical analysis.

After obtaining a benchmark for the 'original' part, several versions of 'attacked' parts were manufactured with modifications at the 1, 2, and 3 mm levels. The reason that these dimensions were chosen is due to the work in [27] that embedded voids with edges approximately 2.5 mm in length. Values in that range were chosen as benchmarks for a detectability threshold:

- *Internal voids*: The same 30x30x1 mm part as the benchmark was created, but with a triangular void of different edge lengths embedded in the center (Figure 4.6a).
- *Scaled parts*: In order simulate a realistic attack, while the lengths changed to 31, 32, and 33 mm, the widths shrunk proportionally to keep the total layer area at 900 mm^2 (30x30

mm). This is a technique that an attacker might use to keep the amount of material usage the same while still altering the part (Figure 4.6b).

- *Skewed parts*: In this attack, one edge of the square layers was moved sideways, also leaving the areas unchanged (Figure 4.6c).

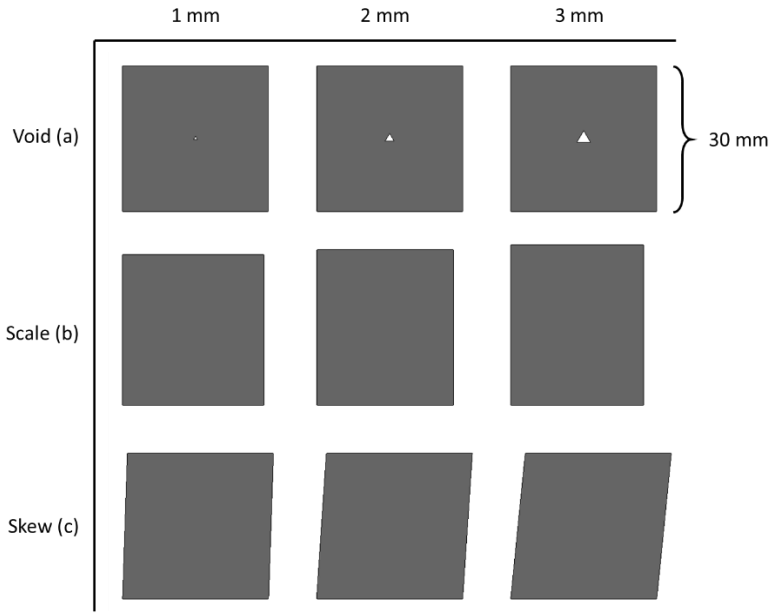

Figure 4.6. Layer cross-section of each STL attack vector and level tested.

4.3.2.2 Experiment 2 – Entire Part

After testing the feasibility of the layer time method, the next step was applying the technique to an entire build. In practice, the time for each individual layer should not matter as much as the overall time for the build. If a layer within the part is modified, then the total time required will likely be modified as well. Experiment 1 was expanded on by manufacturing (and timing) an entire part at once instead of layer by layer. The advantage of this is that less effort is required to count the number of frames recorded by the measurement system. The only disadvantage is that if an attack happens in the middle of the build, the detection will not occur until the build has completed, wasting time and material. The part that was tested is a scaled down

and simplified model of an impeller measuring 30x30x10 mm, shown in Figure 4.7. Each of the fins for the benchmark part have a constant thickness of 1 mm.

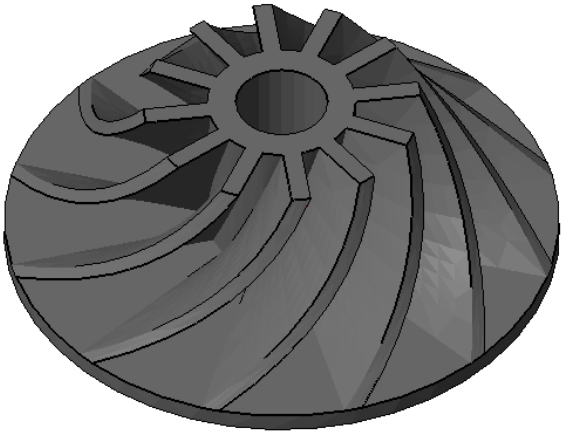

Figure 4.7. Benchmark impeller model. The fins have a constant 1 mm thickness.

Once an average and standard deviation were obtained for the benchmark impeller, slight modifications were introduced to the part to determine the effect of geometry changes on the total build time. The change that was made was thickening the base of each fin. The thickness was changed from 1 mm for the benchmark to 1.25, 1.5, and 2 mm, shown in Figure 4.8. By comparing the build times for each of the modifications to that of the benchmark, this experiment can be considered a success if there is a statistically measureable difference between the two values.

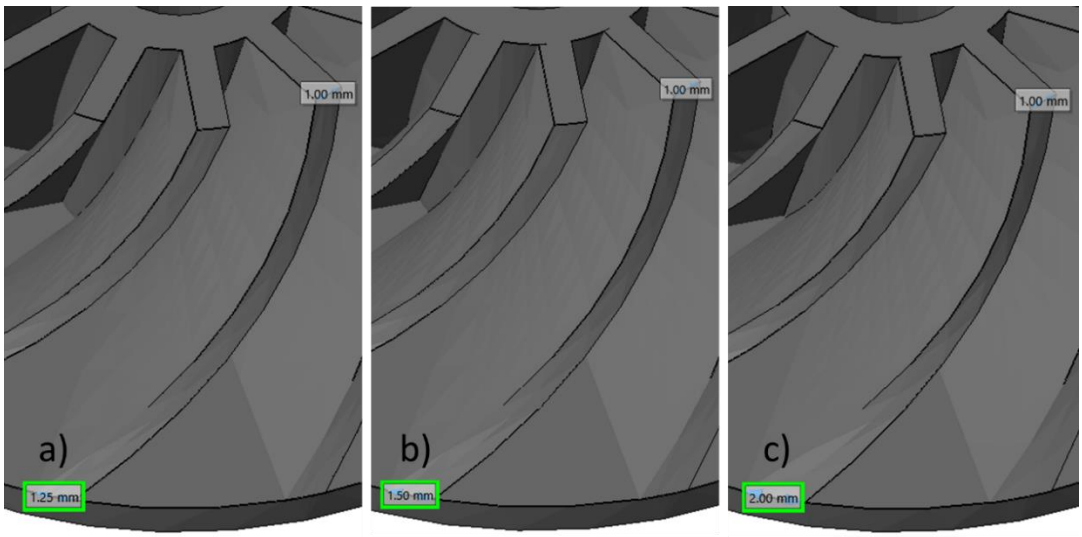

Figure 4.8. Illustration of each modification level for the impeller. The thickness at the base of each fin increases from 1 mm for the benchmark to a) 1.25, b) 1.5, and c) 2 mm.

4.3.3 Layer Time Assessment Results

4.3.3.1 Experiment 1 – Individual Layers

The first major observation that was made is that the time required to build each layer is highly dependent on the raster orientation. In general, the infill pattern employed by most extrusion machines rotates by 90° every layer to help with structural integrity (illustrated in Figure 4.9). This, combined with a 'travel' move that will be discussed later, also causes the time per layer to alternate up and down, shown in Figure 4.10 for the benchmark part.

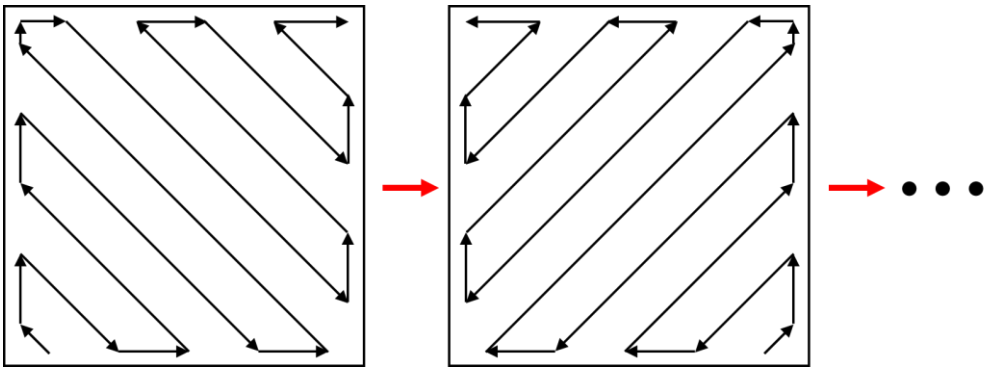

Figure 4.9. Illustration of raster path alternating by 90° every layer.

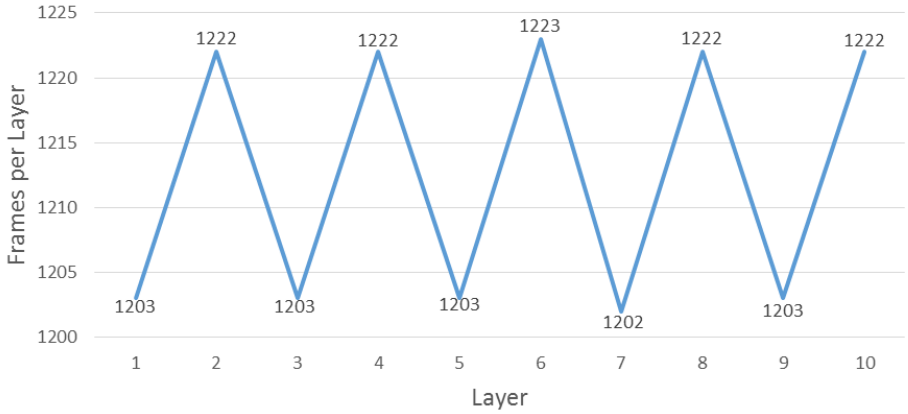

Figure 4.10. Visualization of how the time per layer changes across identical layers. The cause of this alternating effect is that the raster direction rotates by 90° every layer and introduces a 'travel' move.

The fact that the layer times change so much for even and odd layers is not itself significant; it simply means that when building the same layer, two effectively independent averages are obtained. This is irrelevant when applied to the physical hash, because the only values that the

measurement system needs are the upper and lower limits on a range. It does not need to know the orientation at which the layer raster in question was manufactured; it only needs a range to hash for each layer. However, this bimodal distribution does add an extra step to this experiment. Instead of comparing the average of the benchmark times directly to the average of each level of STL attack, two comparisons are made per attack: one for even layers and one for odd layers.

By manufacturing the three types of STL attacks at all three levels, the parts in Figure 4.11 were created. As with the benchmark, each part was recorded at 30 fps and the footage was then used to count the number of frames per layer.

Figure 4.12 shows the average number of frames for the benchmark and the three levels of void edge length. Figure 4.13 shows the same data for the scaled parts, and Figure 4.14 for the skewed parts. The error bars on the plots show ±1 standard deviation based on the five trials (10 layers / 2 independent averages). The raw data that was used to generate these plots can be found in Appendix B.

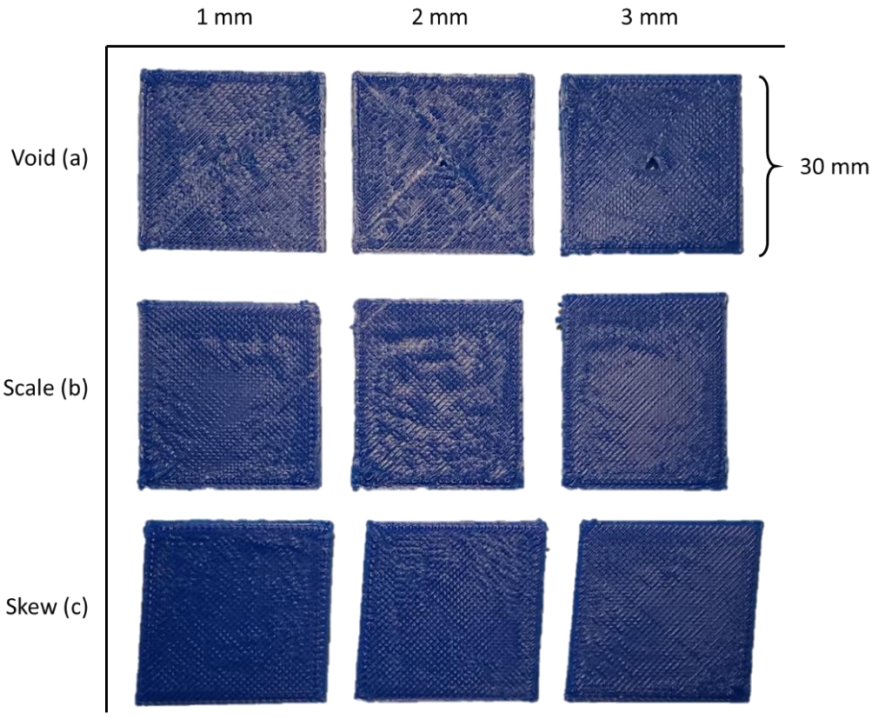

Figure 4.11. Images of the three types of STL attacks and levels after manufacturing.

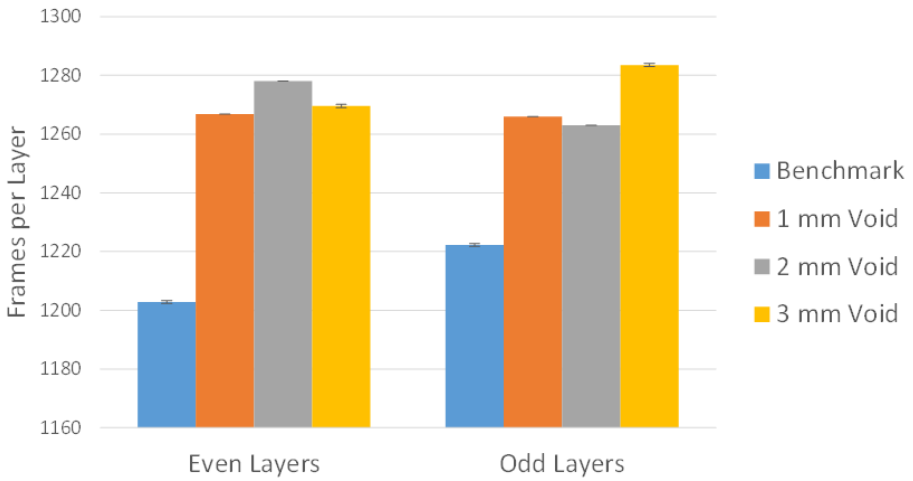

Figure 4.12. Void insertion attack data. The error bars represent one standard deviation based on five trials.

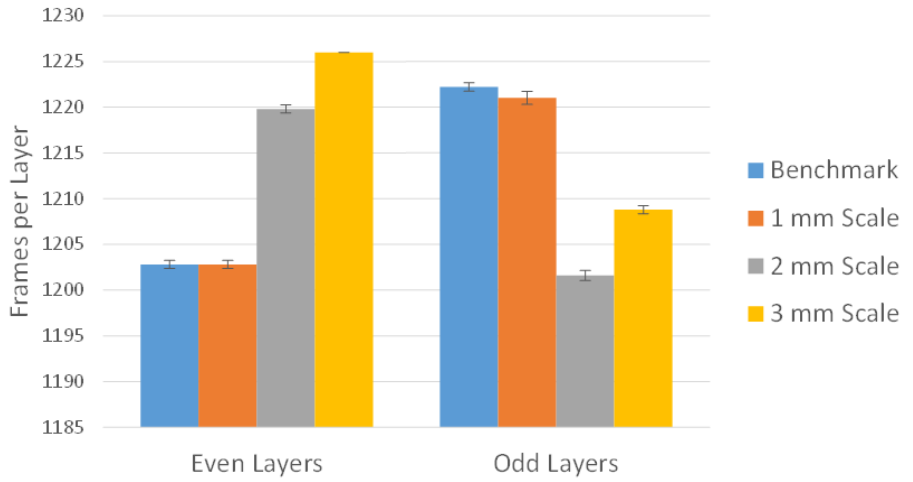

Figure 4.13. Scaling attack data. The error bars represent one standard deviation based on five trials.

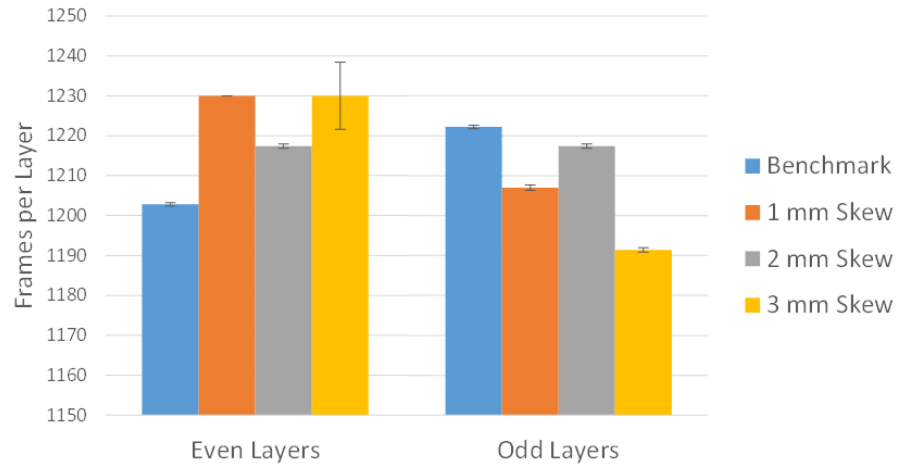

Figure 4.14. Skewing attack data. The error bars represent one standard deviation based on five trials.

The largest change between the benchmark and the frame count at the 1 mm level occurred for the void insertion attack. This was expected because the triangular addition is not present in the benchmark, thus causing a relatively drastic toolpath change. This is different from the scaled or skewed parts, where the extruder head only has to move a bit farther in the same general pattern. While voids can be very difficult to detect with traditional measurement techniques, this result is very promising because even a 1 mm void can alter the toolpath enough to register as a statistically different layer.

The criteria for 'detectability' is that there is a statistical difference between the benchmark and a defect for *both* the even and odd layers. In contrast to the void insertion attacks, the scaled parts do not show enough of a difference between the benchmark and the 1 mm change to be considered detectable. The odd layers do, in fact, show a statistical difference ($p=0.0128$), but the even layers actually averaged the exact same frame count and had the same standard deviation ($p=1$). The smallest scaling attack at which both the even and odd layers show a statistical difference to the benchmark is for the 2 mm change. Thus, the conclusion is drawn that 2 mm is the threshold of detectability for a scaling attack.

It is interesting to note that at the 2 and 3 mm levels for the scaling attack, it almost appears as though the times for the even and odd layers have flipped. However, this is purely a product of the particular toolpath generation algorithm that was used. By analyzing video of the build and comparing it to the measured times, it was discovered that the main difference between even and odd layers comes from an extra 'travel' move that the nozzle completes on certain layers. Travel is when the nozzle moves to a different location without extruding filament. Part of the reason for this type of move is to minimize the time of the overall build. In this case, the software that was used in the experiment determined that for the benchmark and 1 mm levels, it was more efficient

to have a travel move on the odd layers, and for the 2 and 3 mm levels, it was put on the even layers instead.

One may wonder if this travel move would not happen if the pieces were manufactured individually instead of all nine samples at once, but it still does. The important takeaway is not that the times do not follow an obvious trend, but that they are consistent. This is important because it means that this method for detecting modifications is slicer-agnostic. It will work for any toolpath generation software that is consistent for the same settings and part file. Despite this explanation, one can look at the scaling data and ignore the effects of the travel move by comparing the *even* benchmark to the *odd* 2 mm defect, and the *odd* benchmark to the *even* 2 mm defect (and the same for the 3 mm defect). In this analysis, the results still show that the samples are very statistically different. The closest samples are the even benchmark and the odd 2 mm defect, which yield a p value of only 0.0054.

For the skewing attacks, the smallest change is detectable at the 1 mm level, same as for the void attacks. The same 'flipping' effect is seen between the benchmark and 1 mm attack as discussed for the scaling attacks, but even by comparing the closest times (even benchmark/odd 1 mm) to each other, the p value is still less than 0.0001 – extremely statistically significant. There is, however, one anomaly for the skewed attacks that occurs on even layers of the 3 mm part. All of the recorded times over the five samples are either 1233 or 1234 frames, except for one outlier at 1215 frames. By analysis of the video stream it was found that the reason for this change is another travel move, but it is odd that it occurred on only one of the layers. Even in initial exploratory tests when samples were manufactured individually and not in the same batch, this did occur sporadically. While this does throw off the data at the 3 mm level, it is again a reason to point out that should this have been a real part with unique layers, the actual time that each layer

takes to build is not important. The only metric that matters for this method's success is the consistency of the toolpath generation. If the software that is used to generate the benchmark always returns the same frame count for an *entire* part (which this experiment has shown that it should), then when the same software is used to generate an attacked part, there will be a detectable difference in build time.

A similar experiment was tested on a PBF system as well, but the resolution and framerate were both too low to acquire any meaningful data. The framerate was increased to 120 fps, but the laser simply moved too quickly compared to the extrusion system to accurately flag the start and stop times of each layer. Just like with extrusion, the laser in the PBF system follows a set toolpath, so it is assumed that with a high-speed camera, this would be possible as well. The toolpath used on PBF systems generally follows a different path than that used in extrusion, so the trends over increasing defect levels may be different than those discussed above. It is also possible that for layers of equal area, the time does not change at all, making this method impractical for scaling or skewing attacks on PBF. This hypothesis should be tested in future work.

4.3.3.2 Experiment 2 – Entire Part

As a follow-up to the first experiment determining the feasibility of the timing method, an entire part was also tested to see if the technique can scale up from a single layer to an entire part. Other than the benchmark, three levels of fin base thickness were tested: 1.25, 1.5, and 2 mm, shown in Figure 4.15. It is hard to tell from the images that the thickness of the fin base increases, but recall that that is the point. These attacks are modifications that would be very difficult to detect visually but would have an impact on the part's structural integrity.

Figure 4.15. Images of the a) benchmark and three levels of fin base thickness changes: b) 1.25, c) 1.5, and d) 2 mm.

The benchmark (Figure 4.15a) was manufactured four times to obtain an average and standard deviation. Interestingly, even though there was some deviation in the first experiment over the same layer, that error did not compound as expected for an entire part. Instead, the number of frames for the entire build was 49,859 (27.699 minutes) with a standard deviation of exactly zero. This echoes the first experiment by confirming the consistency of the timing method over a build of short to moderate length. The fact that there was no standard deviation means that any change in build time for the modifications can be considered significant. In fact, however, sizable changes in build time were measured between the benchmark and attacked parts. After manufacturing each of the attack levels once, the absolute differences compared to the benchmark were 5.2, 15.1, and 26.4 seconds for the 1.25, 1.5, and 2 mm parts, respectively. A comparison of the frame count for each modification is shown in Figure 4.16 (raw data in Appendix B).

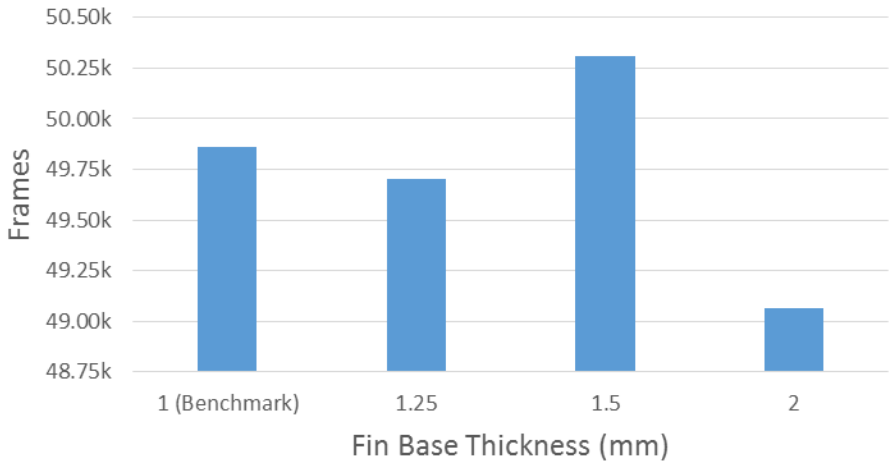

Figure 4.16. Number of frames at 30 fps for the impeller model at the benchmark and three levels of fin base thickness.

4.4 Attacking the Extrusion Temperature Process Parameter

The last research question asks what makes an effective process parameter attack. To answer this question, the effect of a specific parameter, extrusion temperature, on the final part strength was examined in detail. In this section's case study, a rationale and experimental data will be presented for why extrusion temperature is a relevant process parameter that should be monitored *in situ* during extrusion builds.

ABS plastic, a common material used for extrusion, is amorphous and thus "has no true melting point" [61]. Higher end material extrusion systems tend to heat the plastic close to 260 or 270 °C, but many hobby-level machines build as low as 200 °C. This wide range of temperatures is generally seen as an advantage, but the extrusion temperature can affect the strength of an AM part, both in the plane of the layer and in inter-layer adhesion.

Several studies have been done varying the extrusion temperature and found that it has little effect on the final part strength [62], [63]. However these studies only varied the temperature a small amount (270–280 °C in [62] and 280–290 °C in [63]). A real attack would have the ability to change the temperature a great deal more than 10 degrees without significantly affecting the manufacturability of the plastic.

4.4.1 Extrusion Temperature Methods

To investigate if the extrusion temperature can have an effect on the strength of an AM part, a series of dogbone tensile test coupons was manufactured with the dimensions specified in the ASTM-D638-IV standard and a nominal 3.4 mm thickness [64]. The Type IV dimensions were used instead of Type I due to size limitations of the chosen machine. Due to its open-source nature and the ability of the user to directly set the extrusion temperature, the AM system that was chosen for these tests was a custom-built delta-style material extrusion system with an E3D v6 hotend.

The test pieces were manufactured with the main profile in the XY plane with an infill density of 50% (Figure 4.17).

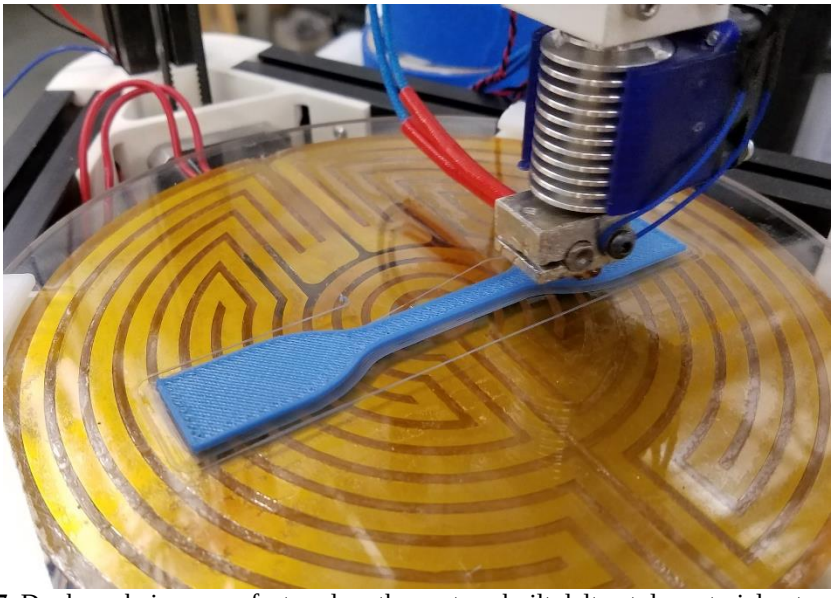

Figure 4.17. Dogbone being manufactured on the custom-built delta-style material extrusion system.

The experiment was carried out at two levels, 235 and 270 °C, with a replication of N=6 for each level. The samples were then pulled on an Instron 5900 Series tensile test machine with a 1 kN load cell and an extension rate of 0.5 mm/min (Figure 4.18).

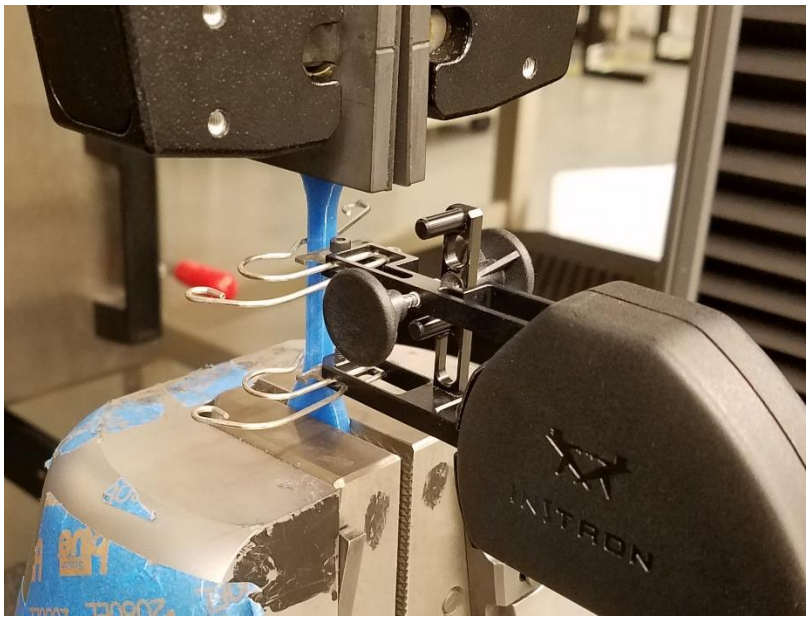

Figure 4.18. Experimental setup for pulling the tensile test specimens on the Instron machine.

4.4.2 Extrusion Temperature Results

After manufacturing six tensile test coupons at the low temperature and six at the high, the parts shown in Figure 4.19 were produced. They were then loaded in the Instron machine and pulled until failure.

Figure 4.19. Tensile test specimens: a) low temp (235 °C) and b) high temp (270 °C).

Two main characteristics of the pull were tracked: Young's modulus and maximum stress. The average values for each characteristic are shown in Figure 4.20. The error bars in each plot show one standard deviation given the six trials. The raw data collected is compiled in Appendix C.

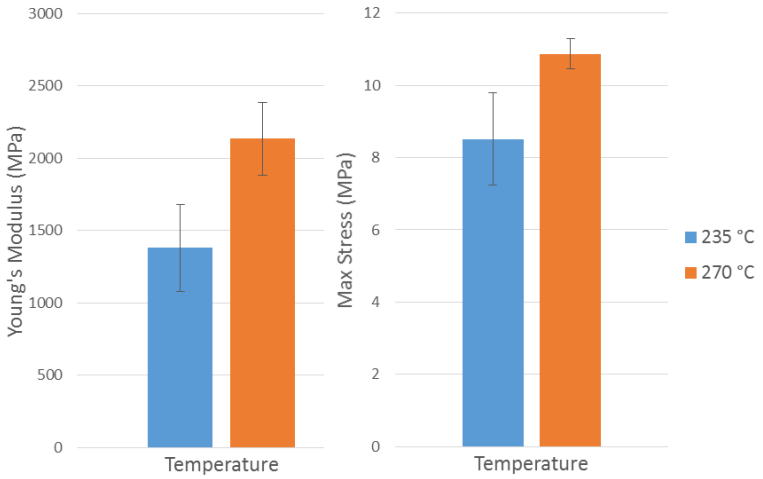

Figure 4.20. Average Young's modulus and maximum stress.

4.4.3 Extrusion Temperature Discussion

From the plots in the results section, it is clear that changing the extrusion temperature by 35 °C had a significant impact on the Young's modulus and maximum stress. From the low to the high temperature, the average Young's modulus over the six samples changed from 1379 to 2135 MPa, and the average maximum stress changed from 8.52 to 10.88 MPa. Testing the hypothesis that the extrusion temperature has an effect on the final part resulted in p values of only 0.0008 and 0.0016 for Young's modulus and maximum stress, respectively. Based on the thresholds $p<0.001X$ and $p<0.01$, there is a very strong statistical difference between the two levels for the Young's modulus and a strong statistical difference for the maximum stress [65].

At the temperature levels tested, there was a noticeable change in surface finish that should be detected in the inspection stage (see Figure 4.21). By holding the part up to the light, it can be seen that the lower temperature sample under-extruded in places due to the plastic not reaching a fully liquefied state. This under-extrusion has been documented in the literature as a result of insufficient heating [43]. An attacker knowledgeable in AM, however, might be able to change the extrusion temperature in combination with other parameters (such as motor torque) to keep a similar bead thickness and surface finish, while still manufacturing the part out of specification. This would visually obscure the fact that a parameter was changed. This experiment successfully showed an example of a process parameter that can have a measurable effect on the final part strength. Recall, however, that extrusion temperature is still only an example of a measurable process parameter. There are many more parameters that can be attacked and may or may not result in a visual change to the part.

Figure 4.21. Comparison of samples manufactured at the low and high temperatures. The lower temperature part (bottom) under-extruded in places, resulting in a worse surface finish.

4.5 Summary of the Physical Hash Approach

In the beginning of Chapter 3, Figure 3.1 showed the overall schematic of the proposed system. The experiments of this chapter have broken that schematic into blocks, and have now made it possible to realize a full implementation of the physical hash methodology.

The QR code blocks of the schematic were addressed in Section 4.1 by verifying that it is possible to use AM to manufacture readable barcodes. Section 4.2 then demonstrated the use of predefined ranges and how they can be used to quantize designed parameters and measurements into 'hashable' tolerance groups. In Section 4.3, a technique for verifying the integrity of the part's toolpath was proposed, enabling the measurement system to rely on benchmark parts as a means of quantizing its measurements into ranges. Finally, in Section 4.4, an experiment was done to measure the effect of extrusion temperature on the final strength of the AM part, motivating the need for the measurement system to monitor certain process parameters. Chapter 5 will now demonstrate a proof-of-concept validation system by drawing on the individual blocks from this chapter.

Chapter 5 – Validation of the Physical Hash Approach

It has been demonstrated that QR codes provide an effective way to transfer information to a disconnected measurement system, and that they can be used in conjunction with predefined ranges in a physical hash. Additionally, a new concept for confirming the integrity of an AM part's toolpath has been developed, and the process parameter of extrusion temperature has been explored as a key parameter in need of *in situ* monitoring. The goal of this chapter is to synthesize these disparate components to validate the overall physical hash approach.

5.1 Experimental Methods

In order to validate that the method described in Chapter 3 performs as expected, it must be able to do three things: a) allow non-attacked parts to be manufactured without a false detection, b) detect a change to a process parameter, and c) detect a change to the toolpath. To test each case, the proposed method was implemented in MATLAB (Appendix A) to allow for (semi)automation of the process. A material extrusion system, the 3DS CubePro Trio, was instrumented with a thermistor and camera filming at 30 fps for this purpose (Figure 5.1).

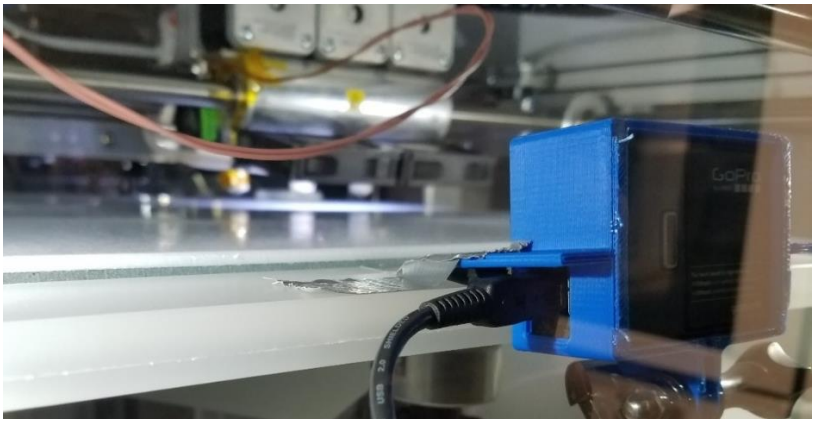

Figure 5.1. Experimental setup for the physical hash implementation. The brown wire leads to a thermistor bonded to the primary extruder. A GoPro HERO3 camera is mounted inside the enclosure pointed at the build tray.

5.1.1 Chosen Side-Channel Measurements

To illustrate how process parameters can be measured on a material extrusion system, a thermistor was bonded to the primary extruder to record extrusion temperatures during the build. Since the thermistor was attached to the outside of the nozzle and not inside the heater block, the values it measured are not the same as the temperature of the molten plastic. The readings are, however, indicative of internal temperatures and should be linearly relatable to the plastic's temperature. This is important because if a malicious attacker targets the extrusion temperature, the change will show up in the side-channel thermistor as well. Extrusion temperature was established as an important parameter to measure in Section 4.4. To measure the toolpath, the time per layer technique developed in Section 4.3 was used to determine if modifications were made to any given layer. A camera mounted inside the enclosure recorded the build and was used to measure how long each layer took to manufacture.

While this implementation combines measurements for both a process parameter and the toolpath, keep in mind that this is only a subset of the possibilities of the proposed system. Any number of sensors can be incorporated into the AM system and used to compare information to the physical hash. This implementation uses only a thermistor and camera as a demonstration of the concept, but in a true industrial-level operation, the chosen sensor suite can and should be capable of measuring any metric of importance, be it a process parameter or relating to the part geometry.

5.1.2 Test Part

The test part that was used for this implementation is a small pyramid with a square cutout (Figure 5.2). The reasoning for this design is that it is simple enough for an AM system to manufacture in only a few minutes, while it still has a couple features that make it a non-trivial build. After manufacturing three benchmark parts to obtain the set of nominal layer times, three

experiments were performed to explore every possibility of the algorithm given this particular sensor suite.

Figure 5.2. Test part used in the material extrusion implementation. The pyramid is 2 mm tall (10 layers) with a 30x30 mm base.

5.1.3 Experiment 1 – Verify Normal Operation

The first experiment was a validation of the working system with a non-attacked part. For this test to be successful, the physical hash should match the hash string generated by the measurement system.

5.1.4 Experiment 2 – Detect an Attack on a Process Parameter

The next experiment attacked the part by changing the extrusion temperature. Unlike the first experiment, this test is successful if the system detects that a malicious modification was made (i.e., the hash strings do *not* match). As demonstrated in Section 4.4, depending on how much the extrusion temperature changes, the final part strength can be greatly affected. Particularly by decreasing the temperature, the plastic may not reach a liquefied state, causing the nozzle to under-extrude. This could result in 'stringier' extrusion beads and weaker interlayer bonds. The change in temperature should cause the measurement to fall in a different predefined range than anticipated by the designer, thus changing the final hash and resulting in a successful attack detection.

The method by which the extrusion temperature was attacked in the second experiment requires a small amount of explanation due to the indirect nature of the attack vector. Unlike the

AM system used in Section 4.4, the machine chosen for this validation experiment limits the user to a single predefined temperature for each material and does not provide an option to directly change it. However, the user is allowed to change the extruder calibration settings, effectively allowing for an indirect attack to be made on its heating element. In order to calibrate the CubePro's heater, the user enters a low and high temperature directly into the machine's LCD panel. These values are used as reference points, between which the internal controller interpolates to accurately control the extrusion temperature. In the factory, the maker of the particular AM system used in these experiments set the calibration values to 218 and 276 °C. However, the interface allows the user (or a malicious attacker in this case) to vary these values anywhere from 214 to 225 °C, and 273 to 284 °C, respectively. With this attack vector, the setpoints were changed from calibration to the highest possible values, 225 and 284 °C (Figure 5.3).

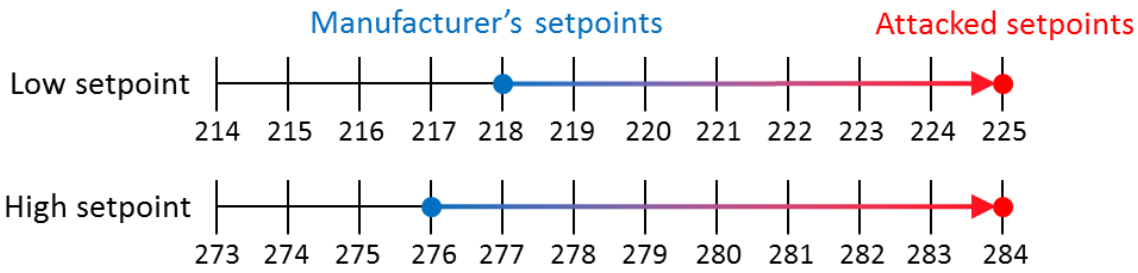

Figure 5.3. Illustration of the attack on the extrusion temperature calibration settings. The blue circles represent the calibration setpoints, and the red circles represent the attacked setpoints.

5.1.5 Experiment 3 – Detect an Attack on the Toolpath

In the third experiment, the toolpath was attacked in two ways. For the first test part, the first attack inserted an internal void into the model. The void was a tetrahedron with 1 mm edges, shown in Figure 5.4a. For a second test part, the second attack indirectly modified the machine code and did not attack the model itself. Instead, the perimeter count or 'outer walls' setting in the toolpath generation code was reduced from four (the default) to two before re-exporting and replacing the file sent to the machine with the attacked version (Figure 5.4b). Like the second

experiment, for both attacked parts, the system should detect the malicious modifications by returning that the hash strings do not match. Both inserting a void or changing the perimeter count setting should cause the affected layers to take a different amount of time to build compared to the benchmark, and the strength of the final parts will be affected as well. When the layer times are measured for the altered parts, some (or all) of them will fall in different predefined ranges than those specified by the physical hash, and the strings will not match.

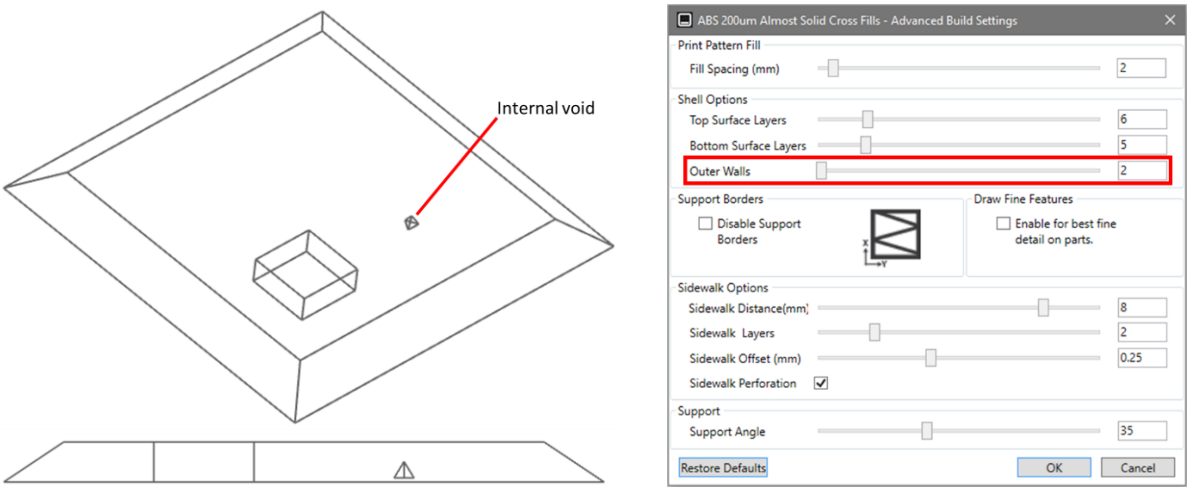

Figure 5.4. a) First attack type: wireframe view of the attacked test part with an internal void maliciously inserted, and b) second attack type: toolpath generation settings with the attacked perimeter count marked.

5.2 Results

5.2.1 Establishing the Benchmark

The process started with layer build time data being collected from three benchmark parts for each of the ten layers. The maximum deviation on any particular layer between the three samples was remarkably only ±1 frame (0.033 seconds), confirming the repeatability of the layer time technique. Based on this information and the allowable tolerance of the extrusion temperature, the designer chose tolerance ‘preset 2,’ which corresponds to range widths of 10 °C for the extrusion temperature and two frames for the layer build times. Just as for the PBF implementation in Section 4.2, the other preset levels define sets of ranges with different tolerances. For example,

'preset 1' is less strict and corresponds to range widths of 15 °C for the extrusion temperature and three frames for the layer build times.

5.2.2 Creating the Physical Hash

The way in which the designer and manufacturer interact with and run the code is very similar to the interactions in Section 4.2. To begin, the designer ran the appropriate script and had a dialog with the program as shown in Figure 5.5.

```
>> Enter tolerance preset (1, 2, or 3): 2
>> Enter the nominal extrusion temperature (°C): 267
>> Enter number of layers with associated build times: 10
>> Enter the nominal build time for layer 01 (frames): 2334
>> Enter the nominal build time for layer 02 (frames): 1412
>> Enter the nominal build time for layer 03 (frames): 1392
>> Enter the nominal build time for layer 04 (frames): 1333
>> Enter the nominal build time for layer 05 (frames): 939
>> Enter the nominal build time for layer 06 (frames): 951
>> Enter the nominal build time for layer 07 (frames): 914
>> Enter the nominal build time for layer 08 (frames): 801
>> Enter the nominal build time for layer 09 (frames): 761
>> Enter the nominal build time for layer 10 (frames): 763
```

Figure 5.5. Dialog for how the designer is asked to input process parameters into script.

By choosing 'preset 2' for the tolerance level, the designer indicated to the program to use the ranges in Figure 5.6. The inputted values were then quantized into these ranges and hashed. To take advantage of the multiple material capability of material extrusion systems, the 3D QR code is not a single file, but rather a separate STL for each color, shown in Figure 5.7.

```
range_extrusion_temp = ...,[240-250),[250-260),[260-270),...
range_layer_time = ...,[2331-2333),[2333-2335),[2335-2337),...

MD5("[260,270],[2333,2335],[1411,1413],[1391,1393],[1333,1335],
[939,941],[951,953],[913,915],[801,803],[761,763],[763,765]")
= "98166ec6bbc574c8845b7a66e610e567"
```

Figure 5.6. 'Preset 2,' the chosen sets of ranges based on the given nominals, and the resulting hash string. The first two entries of the plaintext are color-coded with their corresponding ranges.

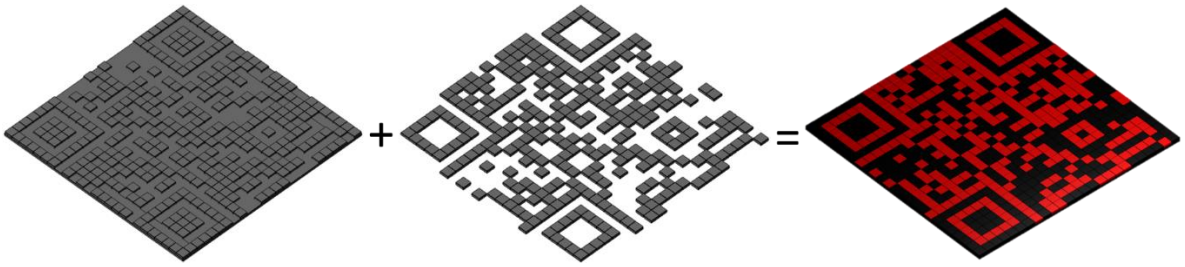

Figure 5.7. a) Dark and b) light regions of the multi-color physical hash. They are manufactured relative to the same coordinate system c) to create a readable part.

5.2.3 Experiment 1 – Verify Normal Operation

To test the ability of the physical hash to confirm normal operation of the AM system, the original STL and physical hash were first manufactured with the default settings of the CubePro Trio. Because the main extruder's heater was turned off when the contrast color of the QR code was being manufactured, steady state was only reached on layers after the code was complete. This is generally not an issue because functional parts will usually have many more layers than are in the QR code. To be considered acceptable, the average temperature for the steady state region must fall in the same range as is in the physical hash.

A plot of extrusion temperature after steady state was reached is shown in Figure 5.8. It may appear from the plot that the temperature is still transient due to the upward shift in temperature at around 100 seconds. However, this shift only occurred due to the extruder-mounted fans that turn on and off at certain points during the build. The region shown in the plot is, in fact, steady state compared to the 'off' temperature which is approximately 80° lower. In this case, the measured average was 266.5 °C, so the range `[260-270)` was selected.

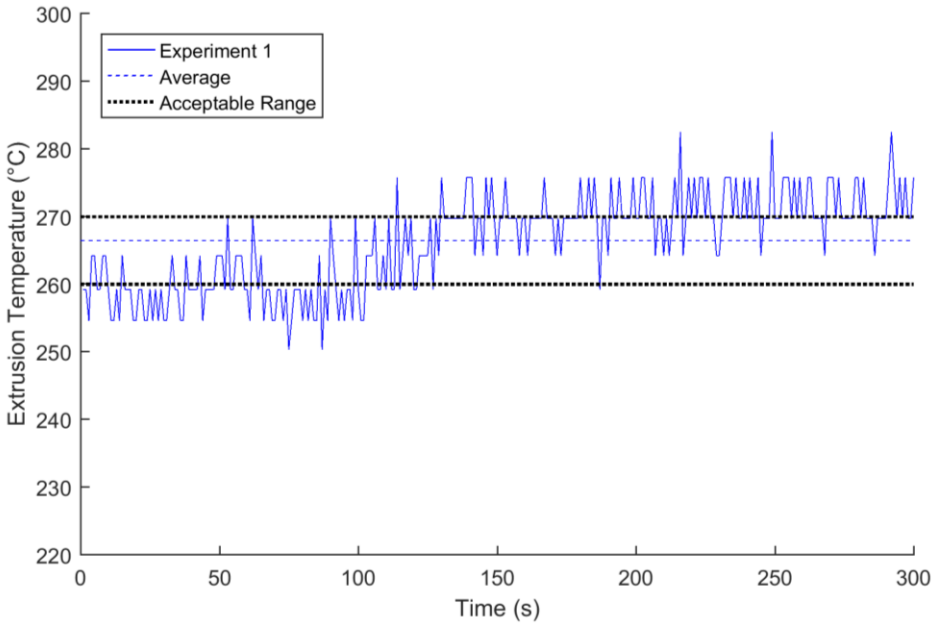

Figure 5.8. Extrusion temperature data after steady state was reached. Note that the average falls in the range `[260-270)`.

The next step was to confirm that the toolpath was not attacked. Figure 5.9 shows a still image from the video stream used to make layer build time measurements.

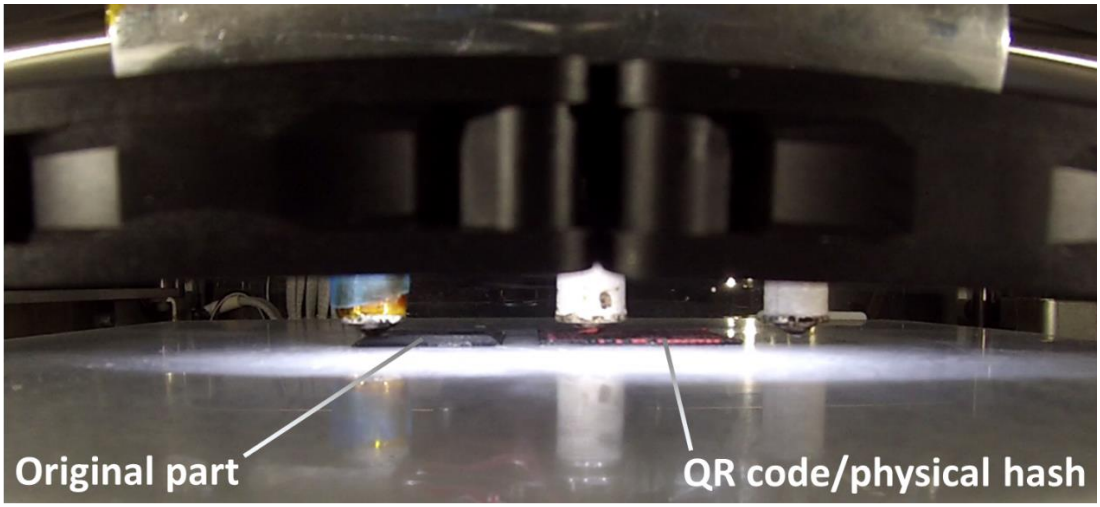

Figure 5.9. Still image from the video stream used to count the number of frames at each layer.

After measuring each layer's build time, the values were quantized into the predefined ranges. For example, 2334 frames were recorded by the camera while the first layer was being manufactured, so the range `[2333-2335)` was chosen. Similarly with the second layer, 1412

frames were recorded so the range `[1411-1413)` was chosen. After this was repeated for each measurement, the layer time ranges were concatenated with the extrusion temperature range and hashed. The QR code was then scanned and decoded (Figure 5.10) and compared to the hash from the measurements. The strings matched (see last line of Figure 5.6), so the method was successful in verifying that an attack did not take place.

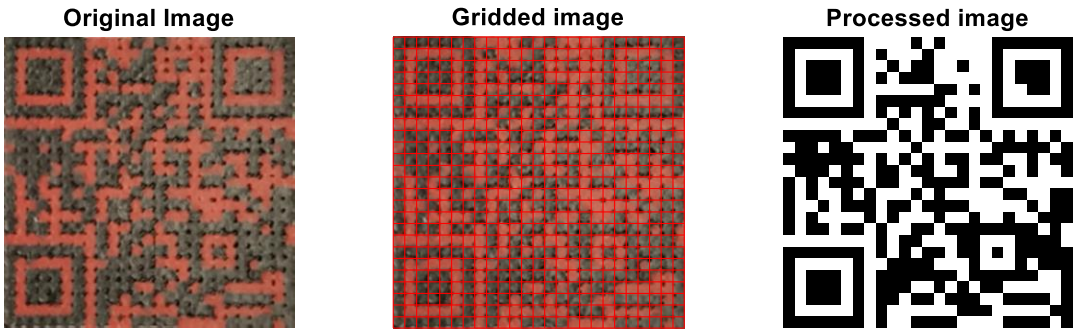

Figure 5.10. Image processing steps for interpreting the QR code: a) original image, b) gridded, and c) final QR to be decoded.

5.2.4 Experiment 2 – Detect an Attack on a Process Parameter

The next experiment tested the capability of the method to detect an attack on the process parameter of extrusion temperature. As a result of the attack, the average temperature dropped from 266.5 to 248.4 °C (Figure 5.11), and the selected range changed from `[260-270)` to `[240-250)`.

Since the toolpath was not modified, the ranges in which the measured layer times fell matched those that were used to create the physical hash. However, since the extrusion temperature range was different, the plaintext that was passed into the hash function changed, as did the resulting hash (Figure 5.12). The hash strings no longer matched, so the attack was detected successfully.

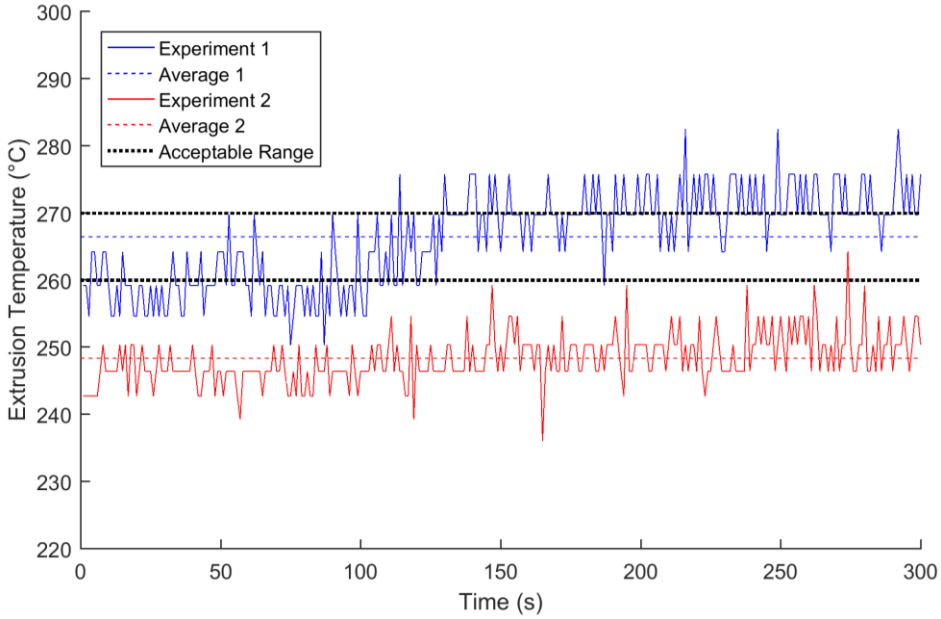

Figure 5.11. Extrusion temperature data after steady state was reached. Note that the average for the second experiment falls outside of the range used to create the physical hash.

MD5("**[240,250]**,[2333,2335],[1411,1413],[1391,1393],[1333,1335],
[939,941],[951,953],[913,915],[801,803],[761,763],[763,765]")
= "**a1ce7f858f74533ca3e2eef02e24ce4f**"
≠ "98166ec6bbc574c8845b7a66e610e567"

Figure 5.12. Plaintext of the measured parameter ranges and the resulting hash string after a process parameter attack. The range corresponding to the extrusion temperature differs from the range used to create the physical hash.

5.2.5 Experiment 3 – Detect an Attack on the Toolpath

In the final experiment, the toolpath of two separate parts was attacked in two separate ways. The first attack modified the model file to contain an internal void. Because the void only appears on certain layers (2–5), only those layers had altered layer times relative to the benchmark, shown in Figure 5.13. The second part was attacked by decreasing the number of perimeters drawn around the part from four to two. Because each layer of the build is different, all of the layer times changed relative to the benchmark, shown in Figure 5.14.

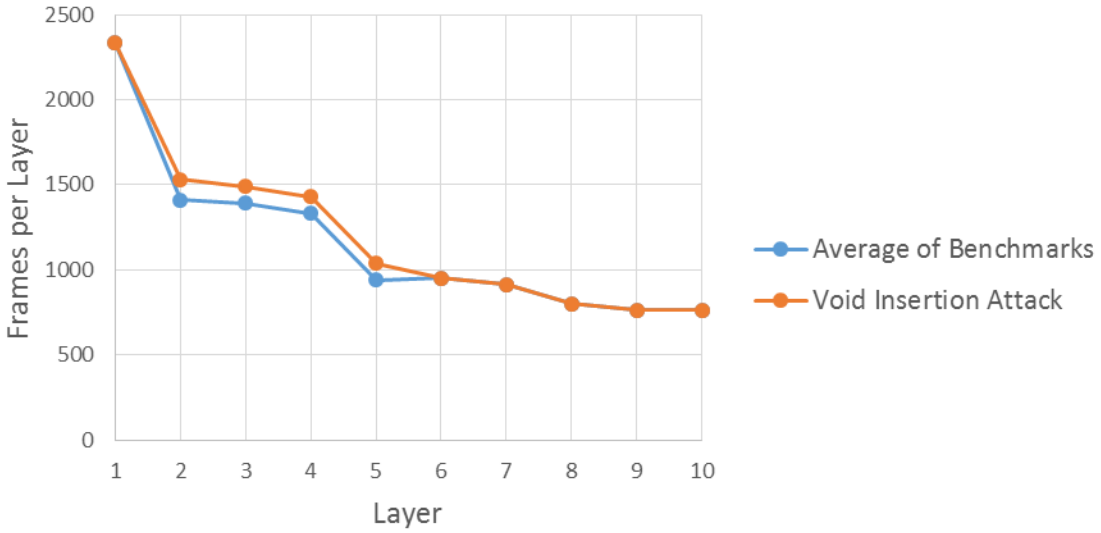

Figure 5.13. Plot of layer times for the benchmark part compared to the part with the void insertion attack. The error bars on the benchmark are only ±1 frame, so they are not visible.

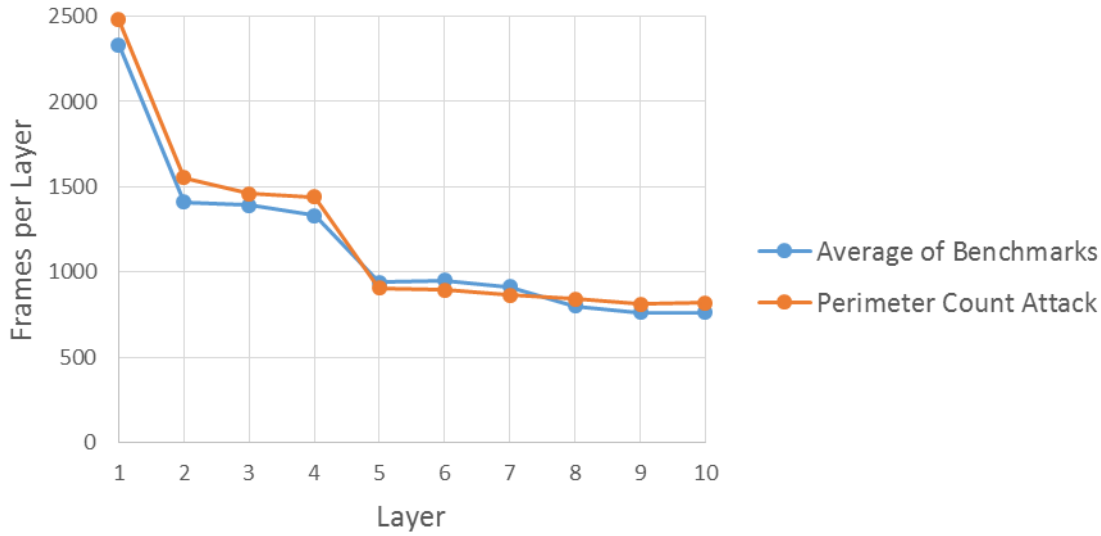

Figure 5.14. Plot of layer times for the benchmark part compared to the part with the perimeter count attack. The error bars on the benchmark are only ±1 frame, so they are not visible.

For both attack types, some or all of the resulting time measurements were quantized to different ranges than those that were used to create the physical hash. The range for the measured extrusion temperature was unchanged in both cases, but since at least one of the layer time ranges was altered, the hash strings changed as well. This resulted in two successful attack detections, show in Figure 5.15 for the void insertion attack, and in Figure 5.16 for the perimeter count attack.

MD5("[260,270],[2333,2335],**[1531,1533],[1491,1493],[1427,1429],**
[1037,1039],[951,953],[913,915],[801,803],[761,763],[763,765]")
= "**c218d8b8605ed08abc3c6dba43b9553a**"
≠ "98166ec6bbc574c8845b7a66e610e567"

Figure 5.15. Plaintext of the measured parameter ranges and the resulting hash string for the void insertion attack. The ranges corresponding to the attacked layer times differ from the ranges used to create the physical hash.

MD5("[260,270],**[2479,2481],[1551,1553],[1459,1461],[1439,1441],**
[905,907],[895,897],[865,867],[843,845],[813,815],[819,821]")
= "**b6a0e9e83793bd367480ddeb52901704**"
≠ "98166ec6bbc574c8845b7a66e610e567"

Figure 5.16. Plaintext of the measured parameter ranges and the resulting hash string for the perimeter count attack. The ranges corresponding to the attacked layer times differ from the ranges used to create the physical hash.

5.3 Discussion

The three experiments presented in this section show that, for the given sensor suite (thermistor and camera), the proposed method successfully confirmed that a non-attacked part passed inspection and that three attacked parts did not. Figure 5.17 shows the parts produced in the three experiments side-by-side. Notice that one cannot tell by eye that the parts are different, although the second was attacked with a modified process parameter and the third and fourth were attacked with modified toolpaths.

Figure 5.17. Image of the parts produced in the three experiments described above: a) no attack, b) attacked extrusion temperature, c) internal void attack, and d) perimeter count attack.

The larger implication this proof-of-concept's success is that the physical hash works as a method of attack detection. This implementation has shown that even if an AM system has been compromised, a disconnected side-channel measurement system can be used to monitor the build in real-time. The physical hash is capable of securely transferring information about the part's process parameters and toolpath without releasing any of the valuable IP it represents.

Chapter 6 – Summary

Additive manufacturing has become increasingly common in the last decade as materials become stronger, machines gain higher and more repeatable resolution, and the technologies within the field become more understood. However, as the manufacturing world becomes more interconnected, cyber-physical systems (such as AM machines) have become vulnerable to malicious cyber-physical attacks.

6.1 Summary of the Physical Hash

In order to protect an AM system from cyber-physical attacks, the only truly secure method is to isolate the machine. This is impractical and a step backwards from the current direction of the industry. Instead, the work in this thesis advocates leaving AM systems connected to the company network and installing a disconnected side-channel measurement system inside the machine. The measurement system is 'air-gapped' from the AM system and company network, but monitors the process parameters and toolpath of the part being manufactured throughout the build.

With the measurement system disconnected, this thesis proposes a *physical hash* as a means of securely communicating to the measurement system the nominals and tolerances against which it should compare its readings. The physical hash is manufactured alongside the original part for the measurement system to scan and interpret. The measurement system can then use its sensor suite to take *in situ* side-channel measurements and compare them to the physical hash to verify that the AM system is operating within specification.

The physical hash takes the form of a QR code and stores a hash string. Hash functions are one-way algorithms and are used in the proposed method to protect valuable intellectual property from a malicious attacker who gains access to an AM system. For every parameter that will be tracked, predefined ranges known to both the designer and measurement system are created that span every possible set of values for that parameter. Given nominal values from the designer, the

ranges in which they fall are selected and hashed before being converted to a QR code (i.e., the physical hash). By measuring values *in situ*, selecting the corresponding ranges, and hashing in the same way as was used to create the physical hash, the measurement system also obtains a hash string. This string can be directly compared to the physical hash, and if they are not equal, the AM system has fallen out of specification.

6.2 Summary of Validation and Results

In addition to proposing the concept of a physical hash, this thesis has also demonstrated its use and practicality with a proof-of-concept implementation. The rationale behind this implementation was built in stages in Chapter 4 to validate the distinct aspects that go into creating the physical hash.

The first element of the overall system to be verified was the use of QR codes as a method of information transfer (Section 4.1). Once it was established that the physical hash could be stored in a QR code and that this idea could be expanded to multiple AM processes, the concept of predefined ranges was explored and validated as well (Section 4.2). Using a powder bed fusion system, it was shown that a single process parameter could be quantized into predefined ranges and represented as a physical hash. A disconnected measurement system then took a reading of that parameter, selected the corresponding range in which it fell, hashed the range, and scanned the QR code to compare hash strings.

Also validated in this thesis is the method of using layer build time measurements to verify toolpath integrity. By measuring the amount of time that a given layer takes to manufacture and comparing to the same measurement on a benchmark part, very small toolpath attacks were able to be detected (Section 4.3). A 1 mm wide void inserted into a 30x30 mm square layer was

identified (Figure 4.12). Additionally, scaling the same layer by 2 mm (Figure 4.13) or skewing it by 1 mm (Figure 4.14) resulted in statistically significant detections.

The importance of the material extrusion process parameter, extrusion temperature, was also validated. By manufacturing dogbone tensile tests at two temperature levels, it was discovered that the extrusion temperature can have a significant impact on final part strength (Section 4.4). This is important because it confirms that this parameter is something that an attacker may decide to target, so the measurement system should be able to detect changes to the extrusion temperature.

These validations of distinct elements of the overall system were combined in a single proof-of-concept implementation (Chapter 5). A material extrusion system was used to validate the ability of the physical hash to a) verify normal operation of the AM system, b) detect an attack on a process parameter, and c) detect an attack on the toolpath. It was demonstrated that the system proposed in this thesis can meet all three criteria without exposing any IP to potentially malicious attackers.

6.3 Contributions

The key contribution of this work to the field is the idea of the physical hash. There is a significant amount of literature motivating why AM systems are vulnerable to cyber-physical attacks, but there is very little work proposing solutions to the problem. While using predefined ranges and hash functions does have certain drawbacks, the method proposed in this thesis is a good first step to a solution and provides a new perspective in the growing field of cyber-physical security.

The second contribution of this thesis is the use of QR codes as a vehicle of information transfer in AM processes. Barcodes have been used as pointers and object tags since their creation, but their use in AM as a checksum verifying integrity of a build has not been explored previously.

This thesis also proposed a method for the side-channel measurement of a part's toolpath involving the amount of time each layer takes to build. This method has been demonstrated to be very robust and repeatable, and adds a new tool to the literature, particularly for material extrusion systems.

6.4 Future Work

This work has introduced a practical method of transferring information from a designer to a disconnected measurement system with minimal involvement of the manufacturer. Additionally, intellectual property is protected during the entire process. While the functionality of the proposed method has been shown via a proof-of-concept implementation on a material extrusion system, there is still work to be done before industry-wide adoption can occur.

The main areas of future work that have been identified can be divided into improvement of the algorithm and improvement of the implementation. For algorithm improvements, the primary drawback of the proposed method is that the predefined tolerance ranges limit the control the designer has over the process. In this work, this is partially solved by allowing the designer to select a preset controlling the width of the ranges, but future study could give rise to modifications in how ranges are defined that avoids this constraint.

The other major area where this work could be expanded is in the implementation. The code written for the proof-of-concept validation managed to automate the majority of the method. However, for the ideas presented in this paper to cross from the lab into industry, a higher level of automation is needed. For example, the layer time measurements were all done by hand, introducing a human element into the method. Additionally, MATLAB is likely not the best programming language to choose for security applications, so work could be done to rewrite the code itself.

Also within implementation improvements, another aspect of the overall method that should be explored in future work is how toolpath on laser-based systems should be measured. The layer time method proposed in Section 4.3 works well for extrusion systems, but did not result in measurements with sufficient accuracy on a PBF system. Other techniques, such as optical or thermal image analysis on a gridded layer image, could replace the layer time method on laser-based systems, but future work is required.

While the validation experiment in this thesis did successfully measure a process parameter and the toolpath, there were essentially only two types of measurements. The next major step for improving on the implementation is to demonstrate that the method can work for a full suite of sensors. Industrial sensor packages already exist commercially, so it would only be a matter of integrating them with the physical hash. This improvement would demonstrate that the physical hash is not just an idea in the lab, but can be practical on an industrial level as well.

In the increasingly interconnected manufacturing and business world, it has become easier than ever for a malicious attacker to hack into a cyber-physical system and have an impact on a company's bottom line or even human life. With the proposed system in place, malicious attackers will have a much harder time of accomplishing their goals.

References

[1] "3D Printing: Facts & Forecasts," *Siemens*, 01-Oct-2014. [Online]. Available: https://www.siemens.com/innovation/en/home/pictures-of-the-future/industry-and-automation/Additive-manufacturing-facts-and-forecasts.html. [Accessed: 21-Feb-2017].

[2] N. Hopkinson and P. M. Dickens, "Analysis of rapid manufacturing-using layer manufacturing processes for production," *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.*, vol. 217, no. C1, pp. 31–39, 2003.

[3] H. Post, "SpaceX Launches 3D-Printed Part to Space, Creates Printed Engine Chamber," *SpaceX*, 31-Jul-2014. [Online]. Available: http://www.spacex.com/news/2014/07/31/spacex-launches-3d-printed-part-space-creates-printed-engine-chamber-crewed. [Accessed: 21-Feb-2017].

[4] M. Balazic, "Additive Manufacturing and 3D Printing LENS Technology," presented at the Additive Manufacturing of Metal Components Conference AT IK4-Lortek, MI, 27-Nov-2013.

[5] J. Hiemenz, "Additive Manufacturing Trends in Aerospace: Leading the Way." 2013.

[6] T. Kellner, "Joined at the Hip: Where the 3-D Printed Jet Engine Meets the Human Body," *GE Reports*, 01-Jul-2013. [Online]. Available: http://www.gereports.com/post/74545196348/joined-at-the-hip-where-the-3-d-printed-jet/. [Accessed: 21-Feb-2017].

[7] M. Yampolskiy, T. R. Andel, J. T. McDonald, W. B. Glisson, and A. Yasinsac, "Intellectual Property Protection in Additive Layer Manufacturing: Requirements for Secure Outsourcing," in *Proceedings of the 4th Program Protection and Reverse Engineering Workshop*, New Orleans, LA, USA, 2014, pp. 1–9.

[8] L. De Bernardini, "Industry 4.0 or Industrial Internet of Things-What's Your Preference?," *Automation World*, 17-Aug-2015. [Online]. Available: https://www.automationworld.com/industry-40-or-industrial-internet-things-whats-your-preference. [Accessed: 21-Apr-2017].

[9] E. Byres and J. Lowe, "The Myths and Facts behind Cyber Security Risks for Industrial Control Systems," *VDE 2004 Congr. VDE Berl.*, pp. 1–6, Oct. 2004.

[10] F. Pasqualetti, F. Dorfler, and F. Bullo, "Cyber-Physical Systems under Attack: Models, Fundamental Limitations, and Monitor Design," presented at the Security Seminar UCLA, University of California, Los Angeles, CA, 24-Feb-2012.

[11] N. Anderson, "Confirmed: US and Israel created Stuxnet, lost control of it," *Ars Technica*, 01-Jun-2012. [Online]. Available: https://arstechnica.com/tech-policy/2012/06/confirmed-us-israel-created-stuxnet-lost-control-of-it/. [Accessed: 21-Feb-2017].

[12] M. Yampolskiy, L. Schutzle, U. Vaidya, and A. Yasinsac, "Security Challenges of Additive Manufacturing with Metals and Alloys," in *Critical Infrastructure Protection IX: 9th IFIP 11.10 International Conference, ICCIP 2015, Arlington, VA, USA, March 16-18, 2015, Revised Selected Papers*, M. Rice and S. Shenoi, Eds. Cham: Springer International Publishing, 2015, pp. 169–183.

[13] M. Holloway, "Stuxnet Worm Attack on Iranian Nuclear Facilities," 16-Jul-2015. [Online]. Available: http://large.stanford.edu/courses/2015/ph241/holloway1/. [Accessed: 21-Feb-2017].

[14] S. Biddle, "(Known) SCADA Attacks Over The Years," *Fortinet Blog*, 12-Feb-2015. [Online]. Available: http://blog.fortinet.com/2015/02/12/known-scada-attacks-over-the-years. [Accessed: 21-Feb-2017].
[15] M. Hagerott, "Stuxnet and the vital role of critical infrastructure operators and engineers," *Int. J. Crit. Infrastruct. Prot.*, vol. 7, no. 4, pp. 244–246, Dec. 2014.
[16] W. J. Broad, J. Markoff, and D. E. Sanger, "Israeli Test on Worm Called Crucial in Iran Nuclear Delay," *The New York Times*, 15-Jan-2011.
[17] P. Marks, "Stuxnet analysis finds more holes in critical software," *New Scientist*, 25-Mar-2011. [Online]. Available: https://www.newscientist.com/article/dn20298-stuxnet-analysis-finds-more-holes-in-critical-software/. [Accessed: 21-Feb-2017].
[18] T. Reed, *At the Abyss: An Insider's History of the Cold War*. New York: Random House Publishing Group, 2005.
[19] A. A. Cárdenas, S. Amin, and S. Sastry, "Research Challenges for the Security of Control Systems," in *Proceedings of the 3rd Conference on Hot Topics in Security*, Berkeley, CA, USA, 2008, p. 6:1–6:6.
[20] "Fraud Alert: Phishing-The Latest Tactics and Potential Business Impacts - Phishing White Paper," *Symantec*, pp. 1–9, 2014.
[21] "Data Breach Investigations Report," Verizon, 2016.
[22] H. Turner, J. White, J. A. Camelio, C. Williams, B. Amos, and R. Parker, "Bad Parts: Are Our Manufacturing Systems at Risk of Silent Cyberattacks?," *IEEE Secur. Priv.*, vol. 13, no. 3, pp. 40–47, 2015.
[23] L. J. Wells, J. A. Camelio, C. B. Williams, and J. White, "Cyber-physical security challenges in manufacturing systems," *Manuf. Lett.*, vol. 2, no. 2, pp. 74–77, 2014.
[24] Z. DeSmit, A. E. Elhabashy, L. J. Wells, and J. A. Camelio, "Cyber-physical Vulnerability Assessment in Manufacturing Systems," *Procedia Manuf.*, vol. 5, pp. 1060–1074, 2016.
[25] T. Fitzgerald and A. J. Bowser, "After explosion, US Department of Labor's OSHA cites 3-D printing firm for exposing workers to combustible metal powder, electrical hazards Powderpart Inc. faces $64,400 in penalties," *Occupational Safety and Health Administration*, 20-May-2014. [Online]. Available: https://www.osha.gov/pls/oshaweb/owadisp.show_document?p_table=NEWS_RELEASES&p_id=26019. [Accessed: 21-Feb-2017].
[26] A. Sternstein, "Things Can Go Kaboom When a Defense Contractor's 3-D Printer Gets Hacked," *Nextgov*, 11-Sep-2014. [Online]. Available: http://www.nextgov.com/cybersecurity/2014/09/heres-why-you-dont-want-your-3-d-printer-get-hacked/93923/. [Accessed: 21-Feb-2017].
[27] L. D. Sturm, C. B. Williams, J. A. Camelio, J. White, and R. Parker, "Cyber-Physical Vulnerabilities in Additive Manufacturing Systems," presented at the SFF Symposium, Austin, TX, 2014, pp. 951–963.
[28] S. Belikovetsky, M. Yampolskiy, J. Toh, and Y. Elovici, "dr0wned - Cyber-Physical Attack with Additive Manufacturing," *CoRR*, vol. abs/1609.00133, 2016.
[29] 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*, Piscataway, NJ, USA, 2016, p. 19:1–19:10.
[30] 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, New York, NY, USA, 2016, pp. 895–907.
[31] "Model Export: VERICUT Module," *CGTech*, 2017. [Online]. Available: http://www.cgtech.com/products/about-vericut/module-model-export/. [Accessed: 21-Feb-2017].
[32] C. Williams, "Direct Metal: Additive Manufacturing, W8D1," presented at the Rapid Prototyping, Virginia Tech, 13-Oct-2015.
[33] M. Albakri, L. Sturm, C. B. Williams, and P. Tarazaga, "Impedance-based non-destructive evaluation of additively manufactured parts," *Rapid Prototyp. J.*, vol. 23, no. 3, 2017.
[34] M. Yampolskiy, P. Horváth, X. D. Koutsoukos, Y. Xue, and J. Sztipanovits, "A language for describing attacks on cyber-physical systems," *Int. J. Crit. Infrastruct. Prot.*, vol. 8, pp. 40–52, Jan. 2015.
[35] M. Yampolskiy, P. Horvath, X. D. Koutsoukos, Y. Xue, and J. Sztipanovits, "Taxonomy for Description of Cross-domain Attacks on CPS," in *Proceedings of the 2Nd ACM International Conference on High Confidence Networked Systems*, New York, NY, USA, 2013, pp. 135–142.
[36] D. Welch and S. Lathrop, "Wireless Security Threat Taxonomy," in *Proceedings of the 2003 IEEE*, West Point, NY, 2003, pp. 76–83.
[37] S. Hansman and R. Hunt, "A taxonomy of network and computer attacks," *Comput. Secur.*, vol. 24, no. 1, pp. 31–43, Feb. 2005.
[38] C. B. Simmons, S. G. Shiva, H. Bedi, and D. Dasgupta, "AVOIDIT: A Cyber Attack Taxonomy," in *9th Annual Symposium*, Albany, NY, 2014, pp. 2–12.
[39] Y. Pan *et al.*, "Taxonomies for Reasoning About Cyber-physical Attacks in IoT-based Manufacturing Systems," *Int. J. Interact. Multimed. Artif. Intell.*, vol. 4, no. 3, pp. 45–54, Mar. 2017.
[40] "National Vulnerability Database," *NIST Computer Security Resource Center*, 20-Mar-2017. [Online]. Available: https://nvd.nist.gov/. [Accessed: 22-Apr-2017].
[41] B. Preneel, "The First 30 Years of Cryptographic Hash Functions and the NIST SHA-3 Competition," in *Proceedings of the 2010 International Conference on Topics in Cryptology*, Berlin, Heidelberg, 2010, pp. 1–14.
[42] N. Yu, "Process parameter optimization for direct metal laser sintering (DMLS)," National University of Singapore, 2005.
[43] T. Pfeifer, C. Koch, L. V. Hulle, G. A. M. Capote, and N. Rudolph, "Optimization of the FDM Additive Manufacturing Process," presented at the SPE ANTEC Indianapolis, Indianapolis, IN, 2016, pp. 22–29.
[44] D. D. Hernandez, "Factors Affecting Dimensional Precision of Consumer 3D Printing," *Int. J. Aviat. Aeronaut. Aerosp.*, vol. 2, no. 4, pp. 1–43, Sep. 2015.
[45] T. G. Spears and S. A. Gold, "In-process sensing in selective laser melting (SLM) additive manufacturing," *Integrating Mater. Manuf. Innov.*, vol. 5, no. 1, p. 2, 2016.
[46] "PrintRite3D - Process Control and Quality Assurance Software for Additive Manufacturing," *Sigma Labs*, 2016. [Online]. Available: https://www.sigmalabsinc.com/products. [Accessed: 22-Feb-2017].
[47] "QM Meltpool 3D," *Concept Laser*. [Online]. Available: https://www.concept-laser.de/en/products/quality-management.html. [Accessed: 22-Feb-2017].
[48] "EOSTATE MeltPool Monitoring," *EOS e-Manufacturing Solutions*. [Online]. Available: https://www.eos.info/software/dmls-meltpool-monitoring. [Accessed: 22-Feb-2017].

[49] I. Wing, R. Gorham, and B. Sniderman, “3D opportunity for quality assurance and parts qualification,” *DU Press*, 18-Nov-2015. [Online]. Available: https://dupress.deloitte.com/dup-us-en/focus/3d-opportunity/3d-printing-quality-assurance-in-manufacturing.html. [Accessed: 22-Feb-2017].
[50] S. Berumen, F. Bechmann, S. Lindner, J.-P. Kruth, and T. Craeghs, “Quality control of laser- and powder bed-based Additive Manufacturing (AM) technologies,” *Laser Assist. Net Shape Eng. 6 Proc. LANE 2010 Part 2*, vol. 5, pp. 617–622, Jan. 2010.
[51] L. Sturm, M. Albakri, C. Williams, and P. Tarazaga, “In-situ Detection of Build Defects in Additive Manufacturing via Impedance-Based Monitoring,” presented at the SFF Symposium, Austin, TX, 2016, pp. 1458–1478.
[52] greener1, “Ember Printer: Bar Codes for Small Parts - All,” *Instructables.com*. [Online]. Available: http://www.instructables.com/id/Ember-Printer-Bar-Codes-for-Small-Parts/. [Accessed: 21-Feb-2017].
[53] K. D. D. Willis and A. D. Wilson, “InfraStructs: fabricating information inside physical objects for imaging in the terahertz region,” *ACM Trans Graph*, vol. 32, no. 4, pp. 1–10, 2013.
[54] M. White, “How Are QR Codes Better Than Barcodes,” *Mobile-QR-Codes.org*, Apr-2014. [Online]. Available: http://www.mobile-qr-codes.org/qr-codes-vs-barcodes.html. [Accessed: 22-Feb-2017].
[55] J. Steeman, “QR Code Data Capacity,” *QR4*, 2017. [Online]. Available: http://blog.qr4.nl/page/QR-Code-Data-Capacity.aspx. [Accessed: 22-Feb-2017].
[56] “QR Code Generator,” *goQR.me*, 31-Oct-2015. [Online]. Available: http://goqr.me/. [Accessed: 26-Feb-2017].
[57] J. Simon, *DataHash*. 2016.
[58] Y. Sasaki and K. Aoki, “Finding Preimages in Full MD5 Faster Than Exhaustive Search,” in *28th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Cologne, Germany, 2009, vol. 5479, pp. 134–152.
[59] J. Motl, *Sauvola local image thresholding*. 2013.
[60] “FDM Materials,” *Forecast 3D*, 2017. [Online]. Available: http://www.forecast3d.com/fdm_materials.html. [Accessed: 22-Feb-2017].
[61] “ABS,” *RepRapWiki*, 15-Apr-2016. [Online]. Available: http://reprap.org/wiki/ABS. [Accessed: 11-Mar-2017].
[62] S. Ahn, M. Montero, D. Odell, S. Roundy, and P. K. Wright, “Anisotropic material properties of fused deposition modeling ABS,” *Rapid Prototyp. J.*, vol. 8, no. 4, pp. 248–257, 2002.
[63] I. Gajdoš and J. Slota, “Influence of Printing Conditions on Structure in FDM Prototypes,” *Tech. Gaz.*, vol. 20, no. 2, pp. 231–236, Apr. 2013.
[64] “ASTM D638-14, Standard Test Method for Tensile Properties of Plastics.” ASTM International, West Conshohocken, PA, 2014.
[65] “P Values (Calculated Probability) and Hypothesis Testing,” *StatsDirect*, 2017. [Online]. Available: http://www.statsdirect.com/help/basics/p_values.htm. [Accessed: 19-Apr-2017].

Appendices

Appendix A – MATLAB Code

The code referenced in this thesis can be found at https://github.com/josheb76/PhysicalHash. The readme file is reproduced below.

```
=====================================================
------------------MAIN EXECUTABLES-------------------
=====================================================
Material Extrusion
- RunMe_Designer_ME.m
- RunMe_MeasurementSystem_ME.m

Powder Bed Fusion
- RunMe_Designer_PBF.m
- RunMe_MeasurementSystem_PBF.m


=====================================================
-------------------STL GENERATION--------------------
=====================================================
Instructions are written to the console when RunMe_Designer_ME/PBF.m is run
- GenerateQR_3D.scad
- GenerateQR_3D_negative.scad


=====================================================
------------------EXPERIMENTAL DATA------------------
=====================================================
Physical measurements used in the implementations in Section 4.2 and Chapter 5

Material Extrusion
- qr_rectified_me.jpg
- extrusion_temperature_data_1.mat
- extrusion_temperature_data_2.mat
- extrusion_temperature_data_3a.mat
- extrusion_temperature_data_3b.mat
- layer_time_data_1.mat
- layer_time_data_2.mat
- layer_time_data_3a.mat
- layer_time_data_3b.mat

Powder Bed Fusion
- qr_rectified_pbf.jpg
- ambient_temperature_data.mat


=====================================================
----------------OTHER MATLAB METHODS-----------------
=====================================================
- averagefilter.m (Credit: [59])
- DataHash.m (Credit: [55])
- DecodeQR.m
- GenerateParameterPlaintext.m
- GenerateQR_2D.m
- GridLayer.m
- LoadPredefinedRanges.m
- Rectify.m
- sauvola.m (Credit: [59])


=====================================================
------AUTO-GENERATED FILES FROM MATLAB/OPENSCAD------
=====================================================
- openscad_formatted_dot_locs.scad
- openscad_formatted_dot_locs_neg.scad
- qr_2D.png
- qr_3D_1.stl
- qr_3D_2.stl
- qr_processed.jpg
```

Appendix B – Time per Layer/Part Experimental Data

Table B.1. Void attack data. Average number of frames per layer are grouped by even and odd layers.

Benchmark	1 mm Void	2 mm Void	3 mm Void
1203	1267	1278	1269
1222	1266	1263	1283
1203	1267	1278	1269
1222	1266	1263	1284
1203	1267	1278	1270
1223	1266	1263	1283
1202	1266	1278	1270
1222	1266	1263	1284
1203	1267	1278	1270
1222	1266	1263	1284

Average	Even Layers	1202.8	1266.8	1278.0	1269.6
	Odd Layers	1222.2	1266.0	1263.0	1283.6
Standard deviation	Even Layers	0.45	0.45	0.00	0.55
	Odd Layers	0.45	0.00	0.00	0.55

Table B.2. Scaling attack data. Average number of frames per layer are grouped by even and odd layers.

Benchmark	1 mm Scale	2 mm Scale	3 mm Scale
1203	1203	1220	1226
1222	1221	1202	1209
1203	1203	1220	1226
1222	1222	1202	1209
1203	1203	1220	1226
1223	1221	1201	1208
1202	1203	1219	1226
1222	1221	1201	1209
1203	1202	1220	1226
1222	1220	1202	1209

Average	Even Layers	1202.8	1202.8	1219.8	1226.0
	Odd Layers	1222.2	1221.0	1201.6	1208.8
Standard deviation	Even Layers	0.45	0.45	0.45	0.00
	Odd Layers	0.45	0.71	0.55	0.45

Table B.3. Skewing attack data. Average number of frames per layer are grouped by even and odd layers.

Benchmark	1 mm Skew	2 mm Skew	3 mm Skew
1203	1230	1217	1234
1222	1207	1217	1191
1203	1230	1218	1215
1222	1208	1217	1192
1203	1230	1217	1234
1223	1207	1218	1191
1202	1230	1218	1234
1222	1207	1218	1192
1203	1230	1217	1233
1222	1206	1217	1191

Average	Even Layers	1202.8	1230.0	1217.4	1230.0
	Odd Layers	1222.2	1207.0	1217.4	1191.4
Standard deviation	Even Layers	0.45	0.00	0.55	8.40
	Odd Layers	0.45	0.71	0.55	0.55

Table B.4. Impeller fin base thickness attack data. Frame counts are for the entire build. Four samples were taken for the benchmark, and one for each modification level.

Trial	Benchmark (1 mm)	1.25	1.5	2
1	49859	49702	50312	49067
2	49859	-	-	-
3	49859	-	-	-
4	49859	-	-	-
Average	49859	49702	50312	49067
Absolute Change (frames)	-	157	453	792
Absolute Change (s)	-	5.2	15.1	26.4
% Change	-	0.315	0.909	1.588

Appendix C – Extrusion Temperature Experimental Data

Table C.1. Temperature change data. The performance metrics are Young's modulus and the maximum stress.

	Trial	Width (mm)	Thickness (mm)	Area (mm^2)	E (MPa)	Max Stress (MPa)
235 °C	1	5.84	3.44	20.09	1533.19	10.20
	2	5.83	3.49	20.35	1810.23	9.62
	3	5.79	3.50	20.27	1465.69	8.84
	4	5.75	3.42	19.67	1011.46	7.93
	5	5.82	3.44	20.02	1073.12	6.76
	6	5.83	3.40	19.82	1382.91	7.74
	Trial	Width (mm)	Thickness (mm)	Area (mm^2)	E (MPa)	Max Stress (MPa)
270 °C	1	5.95	3.54	21.06	1918.31	10.25
	2	5.98	3.51	20.99	2090.85	10.58
	3	6.03	3.48	20.98	2304.70	11.28
	4	5.92	3.57	21.13	2424.54	10.85
	5	5.94	3.51	20.85	2292.45	10.92
	6	5.87	3.37	19.78	1780.29	11.37

		E (MPa)	Max Stress (MPa)
235 °C	Average	1379.43	8.52
	Std Dev	298.63	1.28
270 °C	Average	2135.19	10.88
	Std Dev	249.86	0.42
	p	0.0008	0.0016