# SOK: 3D Printer Firmware Attacks on Fused Filament Fabrication

Muhammad Haris Rais, *Virginia State University;* Muhammad Ahsan and Irfan Ahmed, *Virginia Commonwealth University*

https://www.usenix.org/conference/woot24/presentation/rais

This paper is included in the Proceedings of the 18th USENIX WOOT Conference on Offensive Technologies.

August 12–13, 2024 • Philadelphia, PA, USA

ISBN 978-1-939133-43-4

# SOK: 3D Printer Firmware Attacks on Fused Filament Fabrication

Muhammad Haris Rais[*]
*Department of Computer Science*
*Virginia State University*
*mrais@vsu.edu*

Muhammad Ahsan
*Department of Computer Science*
*Virginia Commonwealth University*
*ahsanm5@vcu.edu*

Irfan Ahmed
*Department of Computer Science*
*Virginia Commonwealth University*
*iahmed3@vcu.edu*

## Abstract

The globalized nature of modern supply chains facilitates hostile actors to install malicious firmware in 3D printers. A worm similar to Stuxnet could stealthily infiltrate a printer farm used for military drones, resulting in the production of batches with a variety of defects. While cybersecurity researchers have extensively delved into the designing and slicing stages of the printing process and explored physical side channels for offensive and defensive research, the domain of firmware attacks remains significantly underexplored. This study proposes a classification tree for firmware attacks, focusing on the attack goals. We further propose nine distinct firmware attacks within these categories to demonstrate and understand the impact of compromised firmware on a standard fused filament fabrication printer. The study evaluates these attacks through relevant destructive and non-destructive tests, including assessing the tensile strength of the printed parts and conducting air quality tests at the printing premises.The study further investigates the viability of forty-eight attacks, including nine that we propose, across the 3D printing stages: the design stage (involving CAD file manipulation), the slicing stage (involving G-code file manipulation), and the printing stage (involving firmware manipulation). Drawing on our understanding of the 3D printing attack surface, we introduce an Attack Feasibility Index (AFI) to assess the feasibility of attacks at different printing stages. This systematization and examination advances the comprehension of potential 3D printing attacks and urges researchers to delve into cybersecurity strategies focused on counteracting feasible attacks at specific printing stages.

## 1 Introduction

The popularity of additive manufacturing (AM) is on the rise [1], with critical industrial sectors such as aerospace [2], automotive, and healthcare [3] utilizing 3D-printed functional parts. Consequently, malicious actors now have greater incentives to attack AM setups and sabotage the printed parts. Concurrently, the current industry trend towards fully connected and converged IT and industrial networks [4] potentially extends the reach of cyber attackers to manufacturing units. Over the past few years, the research community has been actively engaged in both offensive and defensive aspects of AM security. AM is a cyber-physical system (CPS) and as any other CPS technology (SCADA, IOT, etc.) subjected to potential security breaches [5–11].

The existing offensive research focuses on either stealing intellectual property (IP) information through side channels [12, 13] or inducing defects in the printed part [14]. Being fundamentally different from its predecessor technologies, AM or 3D printing (3DP) offers unique attack opportunities for an adversary to sabotage the physical properties of the printed parts. These attacks degrade the object's mechanical strength without modifying the dimensions, weight, center of mass, and other measurable attributes [15]. Although the researchers acknowledge the possibility of firmware attacks [16, 17], most sophisticated attacks are demonstrated at the design and the slicing stages. Moreover, no taxonomy of firmware attacks exists in AM security literature.

This study aims to systematize the knowledge of firmware attacks on 3D printers by developing a classification tree focusing on specific attack objectives. Starting with the fundamental goals of surveillance, denial of service, and integrity breach, the tree extends to include technically attributable subgoals related to the components of the printing process, the printing premises, the printed parts, and the target assembly for which the parts are being printed.

To demonstrate the utility of the classification tree, we showcase nine novel firmware attacks emerging from diverse nodes within the tree, encompassing surveillance, denial of printing service, object integrity, printer damage, and printing premises-related attacks. For instance, *print your own grave* attack prints a tool and uses it to physically damage the printer's components, such as the printing bed. Another attack, named *incurable*, deceives the user by mimicking com-

[*] Rais completed this work while he was a PhD student at Virginia Commonwealth University

mon printing faults, leading to prolonged and ineffective troubleshooting of the printing environment. In a sabotage attack on printing premises, the adversary contaminates the printing facility's environment, leading to potential health hazards. The study demonstrates the impact of these attacks on a standard FFF-based 3D printer running Marlin, the most widely used open-source firmware for 3D printers [18].

The difficulty level of implementing an attack may vary depending on the attack goal. For example, denying printing services by blocking printing instructions is a straightforward task for malicious firmware. In contrast, managing the computation and space complexity involved in scaling an object at the firmware level is challenging. Assessing the complexity of attacks is crucial for understanding the risks involved and prioritizing defensive measures. However, our search revealed no existing research on the complexity analysis of additive manufacturing (AM) attacks.

To fill this gap, we conducted an in-depth analysis of 48 attacks, including those we proposed, to evaluate their implementation complexity at various stages of the printing process. To summarize our findings, we introduce the Attack Feasibility Index (AFI), which represents the feasibility or difficulty level of implementing a specific category of attacks at a particular stage of the printing process.

The findings from the AFI indicate that not all attacks are feasible at any stage of the printing process. For instance, sabotaging the printing premises proves infeasible when targeting the design stage of the process chain. Consequently, cybersecurity solutions optimized for the design stage may not prioritize detecting such attacks.

**Contributions.** This study offers three main contributions:

1. An attack-goal-focused classification tree for the firmware attacks on 3D printers.

2. Implementation and evaluation of nine novel firmware attacks on Marlin-based FFF 3D printer.

3. An analysis of the feasibility of 48 AM attacks at various stages of the AM process chain, providing insights into their implementation complexity at each stage.

## 2 Background and Related Work

### 2.1 Fused Filament Fabrication - FFF

Additive manufacturing, also known as 3D printing (3DP), is a manufacturing technique that constructs objects by adding material layer by layer. This approach fundamentally differs from traditional manufacturing techniques, such as subtractive manufacturing and forging. 3DP offers numerous advantages over previous methods, such as printing complex geometries in a single part, reduced wastage, customized instead of bulk production, and a rapid design-to-production cycle. The ASTM International standards organization defines seven AM methods, including material extrusion, powder bed fusion, and vat photopolymerization, among others [19]. Material extrusion, one of the most commonly used additive manufacturing (AM) techniques, predominantly employs the Fused Filament Fabrication (FFF) process, which is the focus of this study.

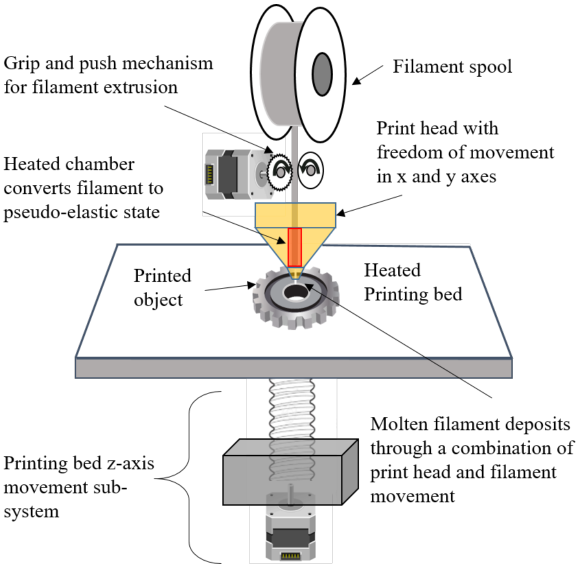

Grip and push mechanism for filament extrusion
Filament spool
Print head with freedom of movement in x and y axes
Heated chamber converts filament to pseudo-elastic state
Printed object
Heated Printing bed
Molten filament deposits through a combination of print head and filament movement
Printing bed z-axis movement sub-system

Figure 1: Fused filament fabrication printer

Figure 1 provides an overview of a typical FFF printer, which creates three-dimensional (3D) objects by extruding molten filament onto a heated printing bed. For first-layer adhesion and to prevent object warping, the printing bed is maintained at a temperature close to the glass transition temperature of the filament. The printing process starts with the solid filament from a material spool fed into the print head by a stepper motor. The print head features a heated chamber that melts the filament into a molten, piezo-elastic state.

The print head utilizes the molten filament extruded from the nozzle orifice to draw a single-layer geometry, resembling a 2D plotter printer with a finite thickness, usually only a fraction of a millimeter. The filament's molten state allows it to pass through the small nozzle orifice, facilitating bonding (fusion) with the previously extruded material to create a solid geometry. Once a layer is complete, the printing bed moves down to create space between the nozzle and the object for the next layer. This layer-by-layer process continues until the desired object is fully printed.

### 2.2 Related Work

This section summarizes current research endeavors focused on firmware attacks targeting material-extrusion AM systems and their classification.

**Attacks Classification.** Several research studies have proposed taxonomies for cyber-physical system (CPS) attacks

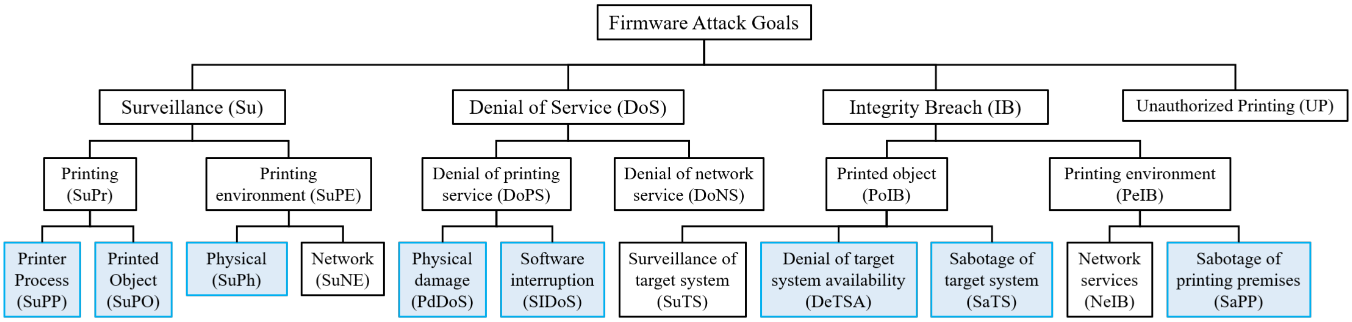

Firmware Attack Goals
Surveillance (Su)
Denial of Service (DoS)
Integrity Breach (IB)
Unauthorized Printing (UP)
Printing (SuPr)
Printing environment (SuPE)
Denial of printing service (DoPS)
Denial of network service (DoNS)
Printed object (PoIB)
Printing environment (PeIB)
Printer Process (SuPP)
Printed Object (SuPO)
Physical (SuPh)
Network (SuNE)
Physical damage (PdDoS)
Software interruption (SIDoS)
Surveillance of target system (SuTS)
Denial of target system availability (DeTSA)
Sabotage of target system (SaTS)
Network services (NeIB)
Sabotage of printing premises (SaPP)

Figure 2: Classification of firmware attacks (categories in blue are covered through proposed attacks)

in AM systems. Yampolskiy et al. [20] developed an attack taxonomy focusing on semantically identical manipulations introduced by compromised elements. Their taxonomy includes a subset of targeted properties known as 'attack targets' but does not delve into the attacker's goals or consider denial of service attacks. In another study [21], they characterize attacks based on manipulated properties.

Pan et al. [22] proposed a taxonomy that comprises vulnerability, attack vector, attack target, and attack impact; however, it is not focused on attack goals. Mahesh et al. [23] presented a four-level attack taxonomy for AM systems, starting with attack goals, methods, targets, and countermeasures. However, their taxonomy does not cover an in-depth attack categorization. Moreover, they see service denial and IP theft as methods rather than attack goals. Wu et al. [24] developed a taxonomy for AM attacks that includes two parallel streams of cyber and physical attacks. It only enumerates a few attack outcomes under cyber and physical attack consequences. Gupta [16] presented multiple supply chain models for an AM process and highlighted the types of attacks associated with those models. They further discussed the risks associated with the studied attacks. The paper summarily mentions the firmware attack vector without delving into the details.

Our study takes a distinctive approach by constructing a multi-tier attack categorization tree based on attack goals within additive manufacturing (AM) systems. This focus allows us to elucidate the strategic intentions behind various attack methodologies and enables the development of targeted countermeasures tailored to thwart specific attack goals.

**Firmware Attacks on 3D Printers.** Researchers have demonstrated sabotage attacks on 3D printers at pre-firmware stages, such as during the design and slicing stages. Additionally, surveillance attacks through side channels have been extensively investigated [25–27]. Nevertheless, exploration into firmware attacks remains relatively limited. Xiao [28] demonstrated firmware attack feasibility on 3D printers. He showcased a thermal manipulation attack by modifying the open-source RepRap firmware through a USB-based serial connection. Moore et al. [29] studied the impact of malicious firmware on print quality by manipulating extruder feed rate or printing alternate geometries. However, their study did not comprehensively analyze the attacks achievable through firmware compromise.

Pearce et al. [30] presented "FLAW3D", a bootloader trojan capable of attacking AVR-based Marlin-supported 3D printers. They demonstrated two low-footprint attacks that could reduce the strength of printed parts. The authors mentioned that only simple manipulation could be feasible due to memory constraints in the bootloader space. Do et al. [31] extracted data from a network-connected printer by exploiting the authentication process vulnerability.

## 3 Classification of Firmware Attacks Goals

**Motivation.** Our motivation for developing the attack classification tree for firmware attacks (Figure 2) stems from the observation that current classifications focus predominantly on the attack actions. Attackers often employ a consistent set of malicious actions at varying intensities to achieve different goals. For instance, low-magnitude thermodynamic manipulation may degrade the object's properties sufficiently for it to fail during operation after installation in the target system. Conversely, high-intensity thermodynamic variations might result in the production of an utterly misshapen object that never gets installed in the target system. In cyber-physical systems, a detection solution might effectively identify actions performed at a higher intensity while failing to recognize those same actions at lower intensities. Consequently, we can classify these detection solutions as effective against one attack category but ineffective against another.

**Methodology.** The methodology involves an iterative division of the attack goals space. Initializing with the top-tier categories encompass surveillance, denial of service, and integrity breach. We also introduce a specific category referred to as 'unauthorized printing,' which pertains to the printing of illegal objects without the process owner's approval. As we move along the tree, the attack goals and the subsequent firmware interactions get more specific. To maintain brevity and comprehensiveness, we limited our exploration to the categories associated with the printed object, the printing process, and the printing environment without delving into further hierarchy. For example, actions resulting in physical damage to any part of the printer, such as targeting the printing bed or nozzle, are encompassed under a single attack goal, denoted as ‘PdDoS.’ The subsequent subsections provide a concise overview of the nodes in the categorization tree.

## 3.1 Surveillance

Surveillance attacks do not modify/sabotage the printing process itself, rather they aim at stealing the printing facility or the printing process information.

### 3.1.1 Printing Surveillance (SuPr)

Printing process surveillance can be further categorized into Surveillance of the Printing Process (SuPP) and Surveillance of the Printed Object (SuPO). The IP information is highly valuable, as its disclosure could result in significant financial losses for businesses. For example, competitors might be interested in gaining insights into the types of prototypes being printed in the research lab. Surveillance can also assist in accomplishing more adversarial goals, such as planning future attacks. In such cases, information pertaining to the network [32], control software and the printer are used to fingerprint the system or design an attack specific to the print geometry.

### 3.1.2 Printing Environment (SuPE)

3D printers can effectively act as spying devices to gather the premises or the environment data. Instead of surveillance of the printing process, the sensing data could be used to illicitly gather information about the facility itself. The information could be from physical sensing systems (SuPh), e.g., cameras, temperature gauges, or network traffic (SuNT), which provide insights into the connected devices over the network. An attacker can use various ways to exfiltrate the information, e.g., by hiding artifacts in the printed object [33].

## 3.2 Denial of Service (DoS)

A malicious firmware can pursue numerous intriguing Denial of Service (DoS) goals by exploiting the digital or physical domains of the printing process. These goals are segmented into two primary categories: Denial of Printing Service (DoPS) and Denial of Network Services (DoNS).

### 3.2.1 Denial of Printing Service (DoPS)

DoPS is accomplished by instigating physical or software disruptions in the printing process. These disruptions may involve causing physical damage to the printer or the printed object, or they can be carried out through software-based attacks aimed at halting or interrupting the printing operation.

**a) Physical Damage (PdDoS).** PdDoS can be achieved by either physical damage to the printer **(PDtP)** or by causing physical damage to the geometry **(PDtG)**. In PDtG, the adversary prints obviously defective or entirely different parts to ensure rejection during post-printing inspection. ‘Print your own grave’ and ‘Incurable’ attacks proposed and demonstrated in Section 5 illustrate these attack categories, respectively. ‘Print your own grave’ prints a tool and uses printing heuristics to damage the printer physically. Conversely, ‘Incurable’ injects observable printing problems in the printed parts to misguide the user into fruitless troubleshooting efforts. These problems include stringing, poor bridging, z-wobble, warping, etc.

**b) Software Interruptions (SIDoS).** A malicious firmware can initiate a DoS attack without physically damaging the printer or the printed part. For instance, setting the command buffer length to zero, triggering an indefinite sleep mode, or circumventing the core printing instructions within the firmware’s main loop can effectively lead to a denial of printing service attack.

### 3.2.2 Denial of Network Services (DoNS)

In this category, malicious firmware conducts traditional Denial of Service (DoS) attacks on networked devices. These attacks encompass network or application-level flooding attacks or exploiting vulnerabilities to crash victim processes.

## 3.3 Integrity Breach (IB)

This category encompasses unconventional attack goals that offer high dividends to the adversary. It is subdivided into two categories based on whether the attack compromises the integrity of the printed object or the printing environment.

### 3.3.1 Printed Object (PoIB)

Printed object integrity breach attacks introduce subtle defects that may find their way through the quality inspection of the target system. The extent of damage depends on the target system instead of the printer. For instance, hidden defects in a 3D-printed car wheel can lead to serious road accidents. This category is subdivided into three types:

**a) Surveillance of Target System (SuTS).** SuTS attacks aim to collect the target system information using the printed object. While no attacks in this category have been demonstrated thus far, the literature suggests the feasibility of incorporating some form of spying capability into a 3D-printed object. For example, printing an RFID tag [34] or watermarking [35] could potentially disclose the location of the printed part.

**b) Denial of Target System Availability (DeTSA).** DeTSA attacks are designed to achieve a denial of service at the target assembly. For example, scaling and axial misalignment attacks, demonstrated in Section 5, are intended to introduce scaling or alignment errors in specific parts, making them infeasible to assemble them into the target system, thereby resulting in target system unavailability.

**c) Sabotage of Target System (SaTS).** While service denial is an additional consequence, SaTS attacks are intended to inflict damage on the target system rather than solely causing a denial of service. Researchers have investigated these attacks in pre-firmware stages [14, 36]. The feasibility of SaTS firmware attacks is demonstrated in Section 5.

### 3.3.2 Printing Environment (PeIB)

Given that the printing environment encompasses digital and physical domains, PeIB is subdivided into two categories.

**a) Network Services (NeIB).** In NeTB attacks, malicious firmware acts as a rogue network element, launching integrity attacks on the computing devices accessible over available networks.

**b) Sabotage of Printing Premises (SaPP).** These attacks physically damage the printing premises, encompassing both the facility and personnel. Through malicious firmware, an attacker can circumvent high-temperature safety controls and exploit filament flammability characteristics to cause a fire hazard [37], or raise the volatile organic compounds (VOC) count in the air to potentially increase the risk of respiratory, cardiovascular, and other disorders [38, 39].

## 3.4 Unauthorized Printing (UP)

Attackers may use malicious firmware to carry out unauthorized printing, such as producing counterfeit goods and manufacturing illegal arms. While inputting static G-code files for these purposes may appear straightforward, the memory limitations present a practical obstacle to launching such attacks using malicious firmware.

# 4 Threat Model

In today's industrial landscape, if malicious firmware infiltrates an FFF printer, it can easily conceal itself from typical printer control software. Operating covertly and evading detection, malicious firmware can enable attackers to execute their objectives for prolonged periods with minimal risk of discovery. An adversary can install malicious firmware through several methods. For instance, researchers have successfully exploited vulnerabilities in printer control software to gain unauthorized access [40]. Once the control software process is compromised, attackers can exploit the printer's standard upgrade routine to install malicious firmware. Additionally, the supply chain offers another avenue for installing such firmware. Brief physical access to the printer during its supply chain journey or while in operation is sufficient to compromise the firmware. This phase follows established tactics observed in previous firmware attacks, as documented in various studies [28, 30, 41–44]. In the subsequent stage, malicious firmware exploits potential vulnerabilities to achieve adversarial goals, as detailed in Section 5.

# 5 Proposed Firmware Attacks

This section presents nine new attacks chosen from the attack categorization tree nodes colored in blue in Figure 2. While the tree encompasses categories related to network surveillance and integrity breaches, we exclusively targeted those emphasizing the specific aspects of the printing process. These attacks are novel at the firmware level, with three previously demonstrated through the manipulation of design files or G-code files. We provide insight into each attack's motivation, the corresponding path within the attack categorization tree, the challenges encountered, the methodology employed, and the outcomes. These attacks are executed on the open-source Marlin firmware, widely utilized in commonly available printers within industrial settings. Specifically, we demonstrated these attacks on the Ultimaker2+ 3D printer.

## 5.1 Object Geometry (OG) Stealing

**Attack motivation.** Stealing the design of a competitor's new prototype offers a significant advantage in time, resources, and market positioning. While prior research has explored IP theft attacks by reverse-engineering the emissions in the physical domain during the printing process [26], our approach distinguishes itself by performing it at the firmware level.

**Attack categorization.** Surveillance → SuPr→ SuPO.

**Outcome.** Stolen geometry of the printed part.

**Challenge.** Marlin firmware running on an embedded system has limited storage, making it infeasible to save the large G-code files that represent printed objects.

**Method.** In this attack, malicious firmware records the potentially useful instructions by developing a small engine that efficiently identifies and captures the sketch of the printed object using three approximations: (1) ignoring the complete infill structure, (2) truncating the sub-millimeter part of x,y coordinates, and (3) activating once per mm of z-axis movement. To address the challenge of identifying the outer shell of the printed object, a circular buffer with sufficient length to accommodate the object's vertices is introduced. The shell is printed at the start or end of each layer, and a shell identification algorithm is employed on the ring buffer at the layer-change event.

One byte is adequate for representing each axis position data in binary format for the case study printer, which has printing-bed dimensions of less than $255 \times 255$ $mm^2$. The engine captures the approximate shape of an object using just 256 bytes and stores it in the EEPROM. As our approach focuses on finding vertices, it is independent of the object's

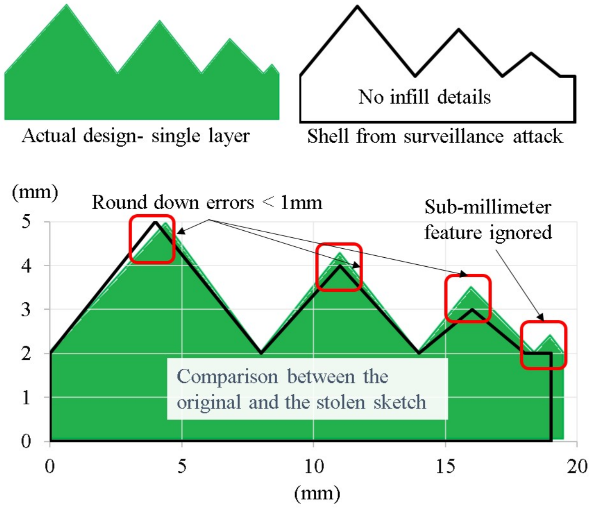


Figure 3: Geometry outline exfiltrated via surveillance attack

size. When an attacker inserts an SD card into the printer, the firmware verifies it and downloads the stolen information within 5 seconds. A variant of this attack can also collect printer hardware configuration information and environment data, such as the ambient temperature, using physical sensors.

**Evaluation results.** Figure 3 illustrates the results of this attack, showcasing an original design (in green), the stolen outline sketch, and an overlaid image highlighting any approximation errors. Despite the omission of sub-millimeter features, the captured sketch provides valuable information to the adversary regarding the object's shape and size, utilizing only 256 bytes compared to the original 32 KB. The attack code disregards infill and solely captures vertices, increasing spatial efficiency with the object's size. For example, a scaled-up version of this object (measuring 10 cm x 2.5 cm x 1 mm) occupies 270 KB of space, while the attacked file still maintains a size of 256 bytes. Furthermore, the percent approximation errors in vertex locations decrease as the object size increases.

## 5.2 Print Your Own Grave (PYOG) Attack

**Attack motivation.** Physical damage to the printer is an effective way to cause denial of printing service (DoPS). In addition to service disruption, the attack entails financial losses incurred from replacing the damaged components.

**Attack categorization.** DoS→ DoPS→ PdDoS→ PDtP.

**Outcome.** A shattered printing bed glass sheet.

**Challenge.** The printing glass is secured through retaining clips over the solid metal sheet. The nozzle is the only other part that comes in contact with the glass sheet. Hitting the nozzle with the bed at maximum speed doesn't provide enough impact to cause damage to the glass.

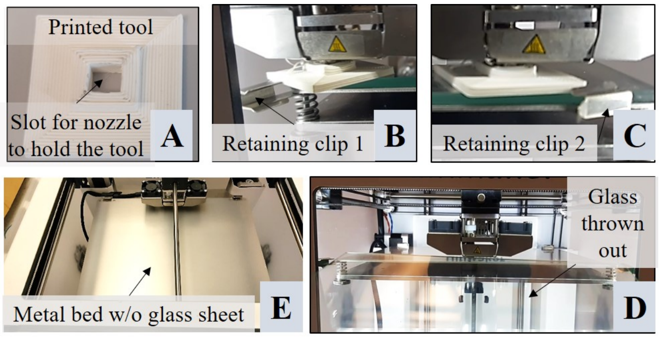


Figure 4: Glass-breaking attack stages

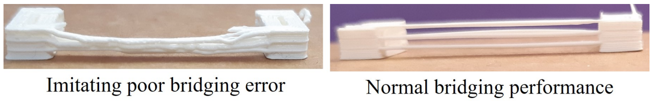
**Method.** PYOG attack exploits the printing function to damage the printer. The presented version of the attack specifically aims at breaking the printing bed glass sheet by throwing it out of the printer. Exploiting the nonexistence of a hardware protection layer between the printing bed and the nozzle, we initially attempted to break the glass by overriding the firmware checks and hitting the bed against the nozzle. The approach, however, does not provide enough impulsive force to break the glass that resides securely over the metal bed. The malicious firmware addresses this challenge by adopting a more sophisticated strategy. It begins by printing a destruction tool, holding it using the nozzle, allowing it to cool down, and then intelligently scans the edges of the printing bed to compromise the glass sheet retaining clips. Finally, the malicious code pushes the glass from the rear edge to throw it out of the printer.

The attack can be triggered by a specific instruction or by an inactivity period. The attack covers two additional categories during execution. The first category is 'software interruption,' achieved by introducing a planned pause and not accepting any printing commands during that time to allow the destruction tool to cool down enough to be detached from the printing bed. The second category is 'unauthorized printing,' which is achieved by printing the destruction tool.

**Evaluation results.** Figure 4 presents a pictorial view of the attack sequence from A to E. The attack utilizes only 20 lines of code to print the tool, normally requiring over 27,000 G-code instructions and more than 500 KB of space. The entire attack code fits well within the available flash memory by only increasing it from 130 KB to 134 KB.

## 5.3 Incurable: Printing Faults Impersonation

**Attack motivation.**Troubleshooting cyber-physical systems, particularly 3D printers, is a laborious and time-consuming task. The rectification and optimization of system configurations necessitate extensive verification through actual printing operations. This motivates us to introduce attacks that mimic known and obvious faulty behaviors, misguiding users into common printing problems and wasting valuable time and



Imitating poor bridging error

Normal bridging performance

Figure 5: Bridging error imitation attack

effort in futile troubleshooting.

**Attack categorization.** DoS→ DoPS→ PdDoS→ PDtG.

**Outcome.** False impression of the poor bridging problem.

**Challenge.** Real-time bridge identification in print geometry.

**Method.** This attack exhibits a poor bridging problem, which tests a printer's ability to extrude filament between two raised points without sagging. An extrusion instruction from $A_{x,y}$ to $B_{x,y}$ in $i^{th}$ layer will belong to a bridge if there is no extrusion between $A_{x,y}$ and $B_{x,y}$ in $(i\text{-}1)^{th}$ layer. To identify a bridge, the attacker must maintain spatial information of the current and previous layers. The attacker cannot analyze and map detailed printing instructions on a compute-constraint system to ensure uninterrupted printing. Hence, the attacker uses a coarse representation of a $100 \times 100\ mm^2$ targeted zone by only a 5 x 5 elements array (named layer-map), where each element represents a square of $20 \times 20\ mm^2$. The bridging performance is typically evaluated over 20 mm and beyond [45]. When a move instruction is received, the layer map is updated, and once a layer is completely printed, it is saved to identify any bridges in the next layer. For each move instruction, the attacker checks if there is any extrusion at the corresponding location in the previous layer. The move instruction is categorized as part of a bridge if there is no extrusion. To create poor bridging performance, the attacker modifies and uses permutation of multiple parameters, including slowing down the cooling fan, increasing the extrusion amount, and reducing printing speed.

**Evaluation results.** We evaluated the attack by printing a shape with three bridges across 25 mm apart pillars, with each bridge added five layers above the previous one. As shown in Figure 5, the attack successfully imitated poor bridging performance. The sag visible on the 25 mm gap between the pillars might mislead users into attributing the poor bridging issue to inefficient printing settings.

## 5.4 Object Feature (OF) Scaling

**Attack motivation.** 3D printing is increasingly used to manufacture critical components for larger assemblies, like turbine blades [46]. If a sub-component of a replacement part is slightly scaled up or down during printing, it will not fit in the assembly, resulting in a delay in service of the target system.

**Attack categorization.** IB → PoIB → DeTSA.

**Outcome.** The attack slightly modifies the dimensions to deny fitment of the printed part.

**Challenge.** This attack goal can be easily accomplished at the designing or the slicing stage using the 'scaling' switch in the software. On the contrary, scaling each printing instruction at the firmware level leads to the scaling of tiny segments connecting consecutive extrudates, which exposes the attack (see Figure 7). Scaling requires additional instructions, and since the firmware receives printing instructions in a temporal sequence, it cannot plan for scaling while printing. Consequently, achieving perfect real-time scaling at the firmware level is not feasible.

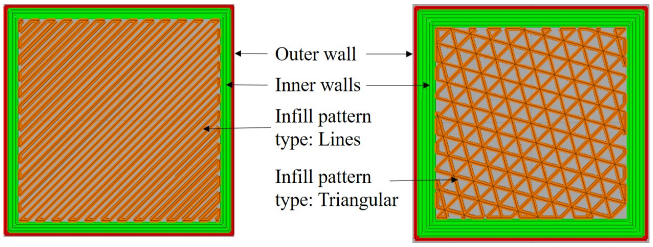


Figure 6: Internal layers composition

**Method.** We exploit the printing format used by slicer software to execute a scaling attack through firmware. A single layer comprises the outer wall structure, the infill pattern for intermediate layers, and skin for the outer layer. Figure 6 shows two infill patterns encapsulated by varying numbers of walls. The outer walls mark the object's edges and create a directed cycle where the destination coordinates for a move instruction are repeated after 'k' instructions (where 'k' represents the number of edges in the object). The wall structure is printed adjacent to the layer change event.

Due to memory constraints, tracking the destination coordinates of all move instructions is not feasible. To overcome this problem, the attacker creates a circular buffer containing one more entry than the maximum number of edges in the anticipated polygon. The firmware searches for a directed cycle to identify a geometrical feature and builds an extra wall around it. Under generic printing settings, wall thickness is proportional to the nozzle diameter, which implies a 0.8 mm to 1.2 mm difference in dimensions across the two opposite walls for 0.4 mm and 0.6 mm nozzles. The attacker uses the change-of-layer instruction to manage the limited computation power to trigger the polygon identification routine. Once the polygon is identified, the attacker selects appropriate coordinates outside the object and prints a new one by adjusting the sequence of the coordinates in the identified cycle.

**Evaluation results.** A rectangular prism with dual sizes was printed to assess the attack's impact. Figure 7 shows a visual comparison between original and attacked samples. While no discernible alterations are evident in the infill structure, the attacked sample exhibits additional walls. We measured the distance between opposite edges at five distinct locations to analyze the dimensional changes. The results indicate an average increase of 0.96 mm ± 0.25 mm for each dimension.

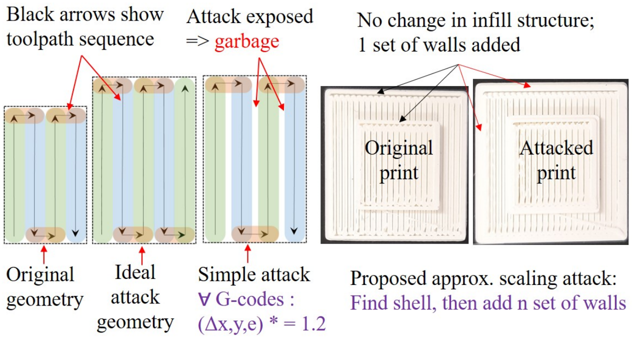

Black arrows show toolpath sequence
Attack exposed => garbage
No change in infill structure; 1 set of walls added
Original print
Attacked print
Original geometry
Ideal attack geometry
Simple attack ∀ G-codes : (Δx,y,e) * = 1.2
Proposed approx. scaling attack: Find shell, then add n set of walls

Figure 7: 2-dimensional object feature scaling attack

## 5.5 Axial Misalignment

**Attack motivation.** The motivation behind this attack is similar to the scaling attack. However, instead of altering the object's dimensions (which are relatively easier to measure), this attack deliberately misaligns a coupling feature over the printing axis to prevent the part from fitting correctly in the target assembly.

**Attack categorization.** IB → PoIB → DeTSA.

**Outcome.** The outcome is an axial misalignment of the coupling slot to deny fitment.

**Challenge.** In addition to the challenges mentioned in the scaling attack, identifying a feature that will ultimately become a coupling candidate, such as a slot, stud, etc., is also challenging.

**Method.** To overcome this challenge, the G-code execution pipeline is delayed by $k_{max}$ printing instructions to ensure that the attack circular buffer as described in Section 5.4 is filled before the printing starts. The malicious firmware searches for a directed cycle within a layer. The temporal distance of the identified cycle from the layer-change event and the length of the constituent lines distinguish between the directed cycles representing a fitment feature and the outer wall structure. Once a change-of-layer event occurs, the x or y coordinates of all vertices of the directed cycle are modified proportionately to the z-axis value to achieve a continuous drift in the feature. This attack targets objects with precise fitment requirements, like driving shafts, assemblies, nuts, and bolts.

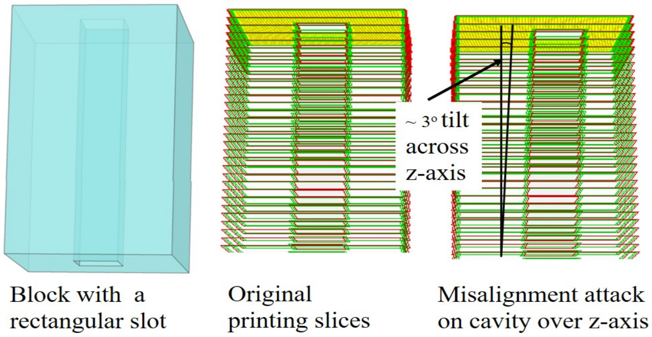

~ 3° tilt across z-axis
Block with a rectangular slot
Original printing slices
Misalignment attack on cavity over z-axis

Figure 8: Geometric feature (coupling slot) misalignment attack

**Evaluation results.** We implemented this attack on a rectangular female square-fitting slot that couples with a male driving shaft. As presented in Figure 8, the attack introduces a $3^o$ axial shift, leading to coupling issues with the male shaft. Unlike Section 5.4, this attack achieves the goal without increasing the number of printing instructions. A variant of this attack only relocates a single vertex of a coupling feature.

## 5.6 Internal Cavity Attack

**Attack motivation.** Reducing an object's strength through hidden cavities is a well-known concern during the design [14] and slicing stages [17]. However, achieving a similar result through malicious firmware remains uncharted territory. The innovation of this approach lies in its execution via firmware. Unlike the digital outputs of the design and slicing stages, the firmware's output during the printing stage is a physical object and is not amenable to standard digital integrity checks. Therefore, the cavity attacks at the firmware stage pose greater risks than those at earlier stages, highlighting the need to investigate firmware-based cavity attacks.

**Attack categorization.** IB → PoIB → SaTS.

**Outcome.** Inducing an internal cavity in the printed part.

**Challenge.** A critical consideration for the success of SaTS attacks is stealthiness. If the attack is exposed, it becomes a DoPS attack. It implies that the cavity should only exist within the internal layers. Deciding on the location and number of layers to induce cavities is an additional challenge.

**Assumption.** This attack is predicated on the assumption that the target object exhibits symmetry along the z-axis. This assumption is valid for ASTM tensile and flexure test models and generally holds for most real-world objects, at least for specific segments of the layer structure.

**Method.** The malicious firmware initially determines the number of G-code instructions in a layer. In the second step, firmware utilizes the G-code instruction 'M73' to identify the candidate internal layers for the attack. In the absence of an 'M73' response, a backup heuristic rule can be used to estimate the number of layers in the object using the standard span-to-thickness ratio of 16:1 recommended by ASTM International Standard [47]. To ensure the cavity remains concealed from the sides within each layer, the attacker splits the instruction into three parts and only stops the filament motor for the central part. The attack ceases once the internal layers are complete, and the printer resumes producing the unaltered top layers.

**Evaluation results.** To evaluate the attack, we printed two ASTM-compliant tensile bars. Figure 9 illustrates the cavity

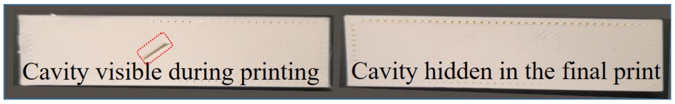

Cavity visible during printing
Cavity hidden in the final print

Figure 9: Cavity attack specimen during and after print

| Sample type | Peak load (N) | | Peak stress (N) | |
|---|---|---|---|---|
| | Avg of 6 samples | Std. dev | Avg of 6 samples | Std. dev |
| Original | 498.44 | 39.65 | 15.38 | 1.17 |
| Attacked | 419.49 | 24.54 | 12.67 | 0.75 |
| **Difference** | **78.96** | | **2.72** | |
| **% Reduction** | **15.84** | | **17.66** | |

Table 1: Tensile test results for filament density attacks

in the left image after pausing the printing process. The top and bottom layers finally cover the cavity.

## 5.7 Object Density Variation Attack

**Attack motivation.** While the cavity in the attack presented in Section 5.6 gets obfuscated in the final object, it is visible during the printing. A more stealthy way to achieve SaTS attack is to reduce the part's density at a critical location. Like cavity, the density variation attack has also been studied at the slicing stage [17]. Hence, the novelty is in its implementation through malicious firmware. Although researchers have achieved generic density variation by attacking the printer [30], our attack is localized with a reduced footprint and improved results.

**Attack categorization.** IB → PoIB → SaTS.

**Outcome.** Reduced object density at a targeted location.

**Assumptions and Challenges.** This attack has the same challenges and the set of assumptions detailed in Section 5.6.

**Method.** The attack preparation steps are the same as those described in Section 5.6, with two changes. Firstly, the zone of interest is increased from one target infill line to a group of lines. Secondly, the retract instructions required for a clean cavity are not included. Instead, the attack manipulates the extruder and filament speed ratio.

**Evaluation results.**We printed six ASTM-compliant tensile bars using the original and attacked firmware and observed no visual or dimensional differences between the two sets of prints. Tensile tests were subsequently conducted using the MTS Insight 30 machine, with the results presented in Table 1. The attacked samples show a 15.84% and 17.66% reduction in the peak tensile load and stress values, respectively.

## 5.8 Filament Erosion Attack

**Attack motivation.** FFF printers estimate the filament quantity using the steps of the stepper motor and the configured filament diameter value. If the filament is partly eroded, the printer will extrude less filament in that region. This motivates us to present a new SaTS attack that erodes the filament to reduce the part density.

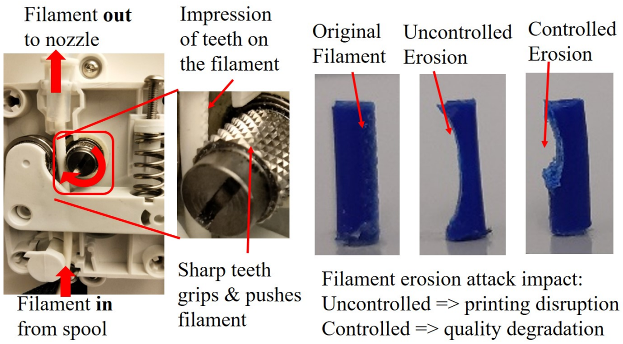

Filament **out** to nozzle
Impression of teeth on the filament
Original Filament
Uncontrolled Erosion
Controlled Erosion
Filament **in** from spool
Sharp teeth grips & pushes filament
Filament erosion attack impact:
Uncontrolled => printing disruption
Controlled => quality degradation

Figure 10: Filament erosion attack

**Attack categorization.** IB → PoIB → SaTS.

**Outcome.** Reduced density of the printed parts.

**Challenge.**While filament erosion may occasionally occur during normal printing operations due to specific routine faults, such as a clogged nozzle, deliberate induction of this phenomenon is not supported by any instructions or functions. If erosion goes beyond a certain point, the printer may not push the filament further, resulting in a denial of service. The challenge lies in creating an erosion function that achieves maximum erosion while maintaining the printer's ability to continue normal printing operations.

**Method.** While the desired outcome is similar to that in Section 5.7, this attack employs an indirect method. In most FFF printers, the extruder motor's shaft teeth grip the filament with the support of a free-rotating roller (see Figure 10). The teeth push the filament axially towards the heated nozzle as the motor rotates. In this attack, a portion of the filament is eroded as it passes through the feeding chamber, reducing the filament quantity at the point of attack. When the defected (eroded) filament portion passes through the nozzle, it creates low-density zones in the printed object. A carefully planned filament erosion attack can thus lower the material density at a critical region, reducing the strength of the printed part. Figure 10 illustrates an example of a filament erosion attack.

The attack uses two methods to erode the filament. The first method involves compelling cold extrusion through the nozzle. Due to the filament diameter being larger than the nozzle orifice, the force exerted by the teeth on the solid filament is insufficient for extrusion through the nozzle. Instead of advancing it, the rotating wheel's teeth merely chip the filament off. The attack bypasses the firmware test routine for the minimum temperature required for extrusion. The second method uses a burst of high-jerk oscillatory movements to break the grooves formed by the gear pressure, resulting in filament erosion. The second method causes less erosion but still achieves the defective printing goal.

**Evaluation results.** We evaluated the effectiveness of the erosion attack using two attack instances. With PLA filament of 2.85 ± 0.1 *mm* diameter, we observed that a cold extrusion motion beyond 1 sec reduced weight from 0.077g to 0.049g, representing a 36% reduction but also interrupting regular operation. Consequently, such an attack could only cause a denial of service. If the attack lasts up to 0.5 secs, the material reduction is up to 20% while regular operation continues, ensuring the required stealthiness to achieve a SaTS attack. The attack employing high-jerk oscillatory move instructions requires additional time to induce material reduction. Conversely, the second method does not necessitate a waiting period for filament cooling, thus offering greater operational flexibility. The second attack resulted in a 15% reduction, with the equivalent length of filament weighing 0.065g.

### 5.9 Printing Facility Air-quality

**Attack motivation.** Given their cyber-physical nature, 3D printers not only facilitate innovative manufacturing processes but also have the potential to negatively impact the physical environment. This reality motivates our investigation into contamination attacks targeting the printer's surroundings.

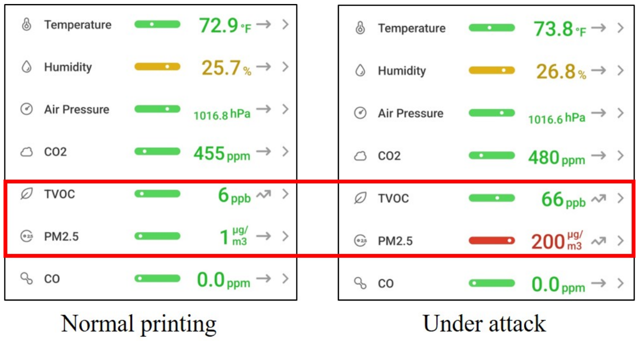
**Attack categorization.** IB → PeIB → SaPP.

**Outcome.** Poor air quality at the printing facility.

**Method.** This attack compromises the air quality in a printing facility by increasing the emission of microparticles and volatile organic compounds (VOCs) through two malicious actions. Initially, it searches for idle state to initiate cold scrubbing bursts at high speeds, which chip fine particles from the filament. Subsequently, the attack manipulates the printer by turning on the nozzle heater while disabling the temperature feedback control circuit. This action leads to the unregulated emission of fumes, VOCs, and microparticles. Due to the elevated temperatures, low-density fluid drips from the nozzle, depositing suspicious droplets on the printing bed. To evade detection, the attacker retracts the filament after leaving only a minimal amount in the chamber and then raises the temperature, which also serves to shorten the attack duration. These actions threaten the environmental health and safety conditions within the facility. Furthermore, the attack may remain undetected with odorless filaments, resulting in prolonged exposure and potential health consequences for the workers.

**Evaluation results.** To evaluate the attack's impact, we conducted experiments to measure the particles and VOC count before and after the attack. Specifically, measurements were taken 5 minutes before the attack and 5 minutes, 30 minutes, and 1 hour after each attack instance. The experiment was repeated five times. We conducted the attacks in a well-ventilated environment with minimum human interaction. Furthermore, no personnel were present in the lab facility to avoid potentially hazardous circumstances.

The findings from these experiments are depicted in Figure 11. The data reveal a significant increase in VOCs from 6 parts per billion (ppb) to 66 ppb, and particulate matter of 2.5 micron or smaller diameter ($PM_{2.5}$) values surged from 1 $\mu$g/m$^3$ to 200 $\mu$g/m$^3$, exceeding the safe limit of 10 $\mu$g/m$^3$ [48]. Due to high ventilation, the recorded values at 30 minutes and one hour after the attack were within normal ranges.



Temperature 72.9 °F
Humidity 25.7 %
Air Pressure 1016.8 hPa
CO2 455 ppm
TVOC 6 ppb
PM2.5 1 μg/m3
CO 0.0 ppm

Normal printing

Temperature 73.8 °F
Humidity 26.8 %
Air Pressure 1016.6 hPa
CO2 480 ppm
TVOC 66 ppb
PM2.5 200 μg/m3
CO 0.0 ppm

Under attack

Figure 11: Air quality stats for facility contamination attack

## 6 Attacks Feasibility/Complexity Analysis

### 6.1 Analysis Methodology

**Motivation.** The attacks described in Section 5 vary significantly in the workload requirements. Depending on the process's stage, the feasibility of initiating an attack can range from trivial to infeasible. For example, object scaling is straightforward at the design stage but becomes complex at the firmware stage. Conversely, thermodynamic attacks are easier to execute through malicious firmware but are unfeasible at the design stage. If an attack is not viable at a particular stage, there is no benefit in implementing defensive measures against it. This prompts us to undertake a comprehensive feasibility analysis of the attack goals (Figure 2) throughout all stages of the printing process chain.

**Methodology.** We begin our analysis by identifying various independent stages in the printing process that attackers could target. We then develop a comprehensive set of feasibility criteria to assign feasibility scores to each of the 48 existing and proposed attacks presented in Table 2. We analyze the proposed attacks using the data presented in Section 5 and draw upon relevant results and findings from the literature on similar attacks.

**Printing process stages.** We analyze the printing process to delineate the independent stages vulnerable to attacks, as illustrated in Figure 12. We treat stages 1a and 1b as a single stage because an attacker who captures the 3D model file (1b) can execute the same attacks by compromising the design software (1a). In contrast, the slicer software (2a), printing profile (2b), and G-code file (2c) each offer distinct capabilities to an attacker, hence considered as distinct stages.

**Feasibility and complexity criteria.** This study employs two factors to evaluate the feasibility of achieving attack goals.

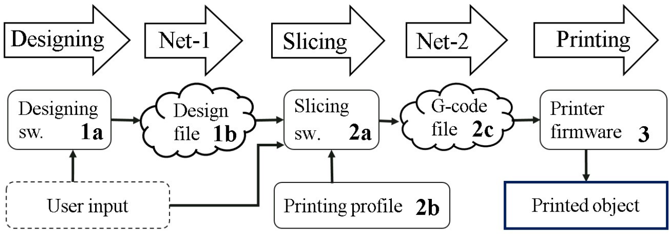


Figure 12: Stages in AM process highlighting cyber artifacts compromisable through a cyberattack

The first factor, $f_1$, assesses the ability to ascertain whether an attack conforms to its intended objective at a specific stage. The second factor, $f_2$, considers the availability of necessary methods to execute attack actions.

For example, at the design stage, the absence of tools to modify the thermal profile in the design file renders dynamic-thermal attacks [17] unfeasible, resulting in an $f_2$ score of zero. In contrast, a simple command can alter the thermal profile at the G-code stage, typically warranting the highest $f_2$ score. However, the firmware's limited temporal perspective complicates the precise placement of the attack, resulting in a low $f_2$ score for these attacks at this stage.

At any particular stage, an attack is considered as:

- an infeasible attack if no execution mechanism or means of confirming compliance with the attack criteria exists.
- a high difficulty attack (●) if both the factors are not readily available and require additional effort to estimate or calculate them.
- a medium-difficulty attack (◐) if one but not both the factors are readily available
- a low-difficulty attack (○) if both the factors are readily available

The feasibility score of the $n^{th}$ attack at the $m^{th}$ stage, $FS_{n,m}$, is defined in Eq. 1 as the product of $f_{1_{n,m}}$ and $f_{2_{n,m}}$, where $f_1$ and $f_2$ are assigned the values of 0, 1, and 2 for 'not available,' 'not readily available,' and 'readily available' respectively.

$$FS_{n,m} = \begin{cases} Infeasible & \text{if } f_{1_{n,m}} \times f_{2_{n,m}} = 0 \\ High\ difficulty & \text{if } f_{1_{n,m}} \times f_{2_{n,m}} = 1 \\ Medium\ difficulty & \text{if } f_{1_{n,m}} \times f_{2_{n,m}} = 2 \\ Low\ difficulty & \text{if } f_{1_{n,m}} \times f_{2_{n,m}} = 4 \end{cases} \tag{1}$$

## 6.2 Attack Analysis

Table 2 outlines the attack actions, literature reference, their type in the attack categorization tree, and the feasibility status in light of the above-mentioned criteria. Due to space constraints, we collectively discuss them under the following subsections. Where, $A_1 - A_{48}$ in the discussion refers to the attack serial number in the table. A short description of each attack is provided in the Table 4 (Appendix A).

**Designing stage.** The designing stage focuses on the geometry of the desired object. The fitment attacks for DeTSA $(A_{25} - A_{27})$, anisotropy attacks $(A_{45})$, and geometric feature insertion or removal $(A_{42})$ for SaTS are the simplest and most accurate at the designing stage. Surveillance attacks on the printed object $(A_4)$ and the connected network $(A_6)$ are also feasible if the attacker has access to the designing software process.

**Slicing and control software.** With an STL file and printing profile as the input and a G-code file as the output, the stage offers a vast spectrum of attack opportunities. It outperforms all other stages in achieving SaTS goals $(A_{29} - A_{47})$. However, the stage is less effective for DeTSA and PDtG attack goals.

**Printing profile.** The printing profile comprises parameters used by the slicer software to attain a set of printing instructions. It can easily launch attacks related to global parameter settings $(A_{17}, A_{34}, A_{35}, A_{37}, A_{38})$.

**G-code file through Net-2.** The chronological structure of a G-code file suits the introduction of localized defects to achieve most of the DeTSA $(A_{25} - A_{27})$, SaTS $(A_{29} - A_{47})$, and PDtG $(A_{14} - A_{23})$ attacks. Infill pattern and density attacks $(A_{34}, A_{35})$, however, are challenging to execute.

**Firmware.** For incurring damage to the printer and facility $(A_9 - A_{13}, A_{48})$, and most of the other PDtG $(A_{14} - A_{23})$, the firmware stage leads all other stages. However, firmware is the second-best stage to launch the DeTSA and SaTS attacks, following the slicer. One reason is the difficulty in achieving the required stealthiness and accuracy due to its limited temporal view.

## 6.3 Attack Feasibility Index - AFI

To assess the feasibility of attack categories at different stages of the printing process, we introduce the term 'Attack Goal Feasibility Index' (AFI), ranging from 0 to 1. An AFI value of 0 indicates that an attack goal is not feasible at a particular stage. In contrast, an AFI value of 1 indicates low difficulty as defined in Section 6.1. The index incorporates the cumulative effect of all attacks in a particular category and is calculated as follows:

$$AFI_{g,s} = \frac{1}{n \times F_{max}} \sum_{i=1}^{n} FS_{i,s} \tag{2}$$

where, $AFI_{g,s}$ is the AFI value for the attack category $g$ at stage $s$. $n$ is the total number of attacks in the category $g$, $F_{max}$ is the numeric value '4' assigned to the 'low-difficulty' level, and $FS_{i,s}$ represents the feasibility score of $i^{th}$ attack at stage $s$.

Table 3 displays the Attack Feasibility Index (AFI) for the examined attacks, broken down by the stages (Figure 12). Only the slicing software (2a) and the firmware (3) stage demonstrate non-zero AFI values across all attack goals. The normalized AFI value for SaTS attacks is 0.89 for Stage 2a

| Sr. No. | Attack Name | Ref. | Attack goal category | Designing (1a/1b) | Slicing control software (2a) | Printing profile (2b) | Net-2 (G-code file) (2c) | Firmware (3) |
|---|---|---|---|---|---|---|---|---|
| 1 | Printer info | [49] | SuPP | - | ◐ | - | ◐ | ○ |
| 2 | Design SW info | [50] |  | ○ | ◐ | - | - | - |
| 3 | Slicer/Control info | [50] |  | ◐ | ○ | - | ◐ | ◐ |
| 4 | OG info | [25,26], *P** (5.1) | SuPO | ○ | ○ | - | ○ | ● |
| 5 | Print profile info | [27] |  | - | ○ | ○ | ○ | ◐ |
| 6 | Network device | [32] | SuNT | ○ | ○ | - | ○ | ○ |
| 7 | Process artefacts | [33] |  | ● | ● | - | ● | ● |
| 8 | Facility info | *P* | SuPh | - | - | - | - | ◐ |
| 9 | PYOG | *P** (5.2) | PDtP | - | ● | - | ● | ◐ |
| 10 | Breaking limits | *P* |  | - | - | - | - | ○ |
| 11 | Nz impair | *P* |  | - | - | - | - | ◐ |
| 12 | Extruder fracture | [28] |  | - | - | - | - | ◐ |
| 13 | Nz burning | *P* |  | - | - | - | - | ◐ |
| 14 | OG scaling | *P** (5.4) | PDtG | ○ | ○ | ○ | ○ | ○ |
| 15 | OG thermal | [28] |  | - | ○ | ○ | ○ | ○ |
| 16 | Incurable | *P** (5.3) |  | - | ◐ | ● | ◐ | ◐ |
| 17 | Warping | [17] |  | - | ○ | ○ | ○ | ○ |
| 18 | FK thermal | [28] |  | - | ○ | - | ◐ | ○ |
| 19 | Trajectory unsync | [51], [28] |  | - | ● | ● | ● | ◐ |
| 20 | PS profile | *P* |  | - | - | - | - | ○ |
| 21 | PS unsync | *P* |  | - | - | - | - | ○ |
| 22 | FK reduction | [30] |  | - | ○ | - | ○ | ○ |
| 23 | Clogging | [28] |  | - | ◐ | ● | ◐ | ◐ |
| 24 | MAC/ARP corruption | [50] | SIDoS | - | ○ | - | - | ○ |
| 25 | Vertex relocation | [52] | DeTSA | ○ | ◐ | - | ◐ | ◐ |
| 26 | OF scaling | *P** (5.4) |  | ○ | - | - | - | - |
| 27 | Fitment | *P** (5.5) |  | ○ | - | - | ◐ | ◐ |
| 28 | NT manipulation | [49] | NeIB | ○ | ○ | - | ○ | ◐ |
| 29 | IF line spacing | [15] | SaTS | - | ○ | - | ○ | ◐ |
| 30 | IF vertex spacing | [15] |  | - | ○ | - | ○ | ◐ |
| 31 | FS maniplation | [17], [40] |  | - | ○ | - | ○ | ○ |
| 32 | FK cavity | [17], *P** (5.6) |  | - | ○ | - | ○ | ◐ |
| 33 | FK density | [17], *P** (5.7) |  | - | ○ | - | ○ | ◐ |
| 34 | IF pattern | [53] |  | - | ○ | ○ | ● | - |
| 35 | IF density | [53] |  | - | ○ | ○ | ● | - |
| 36 | PS cavity | [17] |  | - | ◐ | - | ◐ | ◐ |
| 37 | IF exclusion | [40], [30] |  | - | ○ | ○ | ○ | ○ |
| 38 | % IF | [54] |  | - | ○ | ○ | ◐ | ◐ |
| 39 | GF change | [29], [51] |  | - | ○ | - | ○ | ◐ |
| 40 | Dyn.Thermal | [17], [54] |  | - | ○ | - | ○ | ◐ |
| 41 | PS local | [54] |  | - | ○ | - | ○ | ○ |
| 42 | OF insert/remove | [14], [55] |  | ○ | - | - | - | - |
| 43 | LT local | [54] |  | - | ○ | - | ○ | ○ |
| 44 | GC sequence | [54], [56] |  | - | ○ | - | ○ | ○ |
| 45 | Anisotropy | [55] |  | ○ | ○ | - | ● | - |
| 46 | GC manipulation | [54], [56] |  | - | ○ | - | ○ | ○ |
| 47 | Erosion | *P** (5.8) |  | - | ● | - | ● | ◐ |
| 48 | PF air quality | *P** (5.9) | SaPP | - | ◐ | - | - | ○ |

- : Not feasible ●: High difficulty ◐: Medium difficulty ○: Low difficulty $P/P^*$ : Proposed/Proposed and Evaluated

Table 2: Categorization of existing and proposed attacks with their feasibility and difficulty at various stages

| **Attack Goal Category** | **Normalized attack feasibility index for process stages** | | | | |
|---|---|---|---|---|---|
| | **1.** | **2a.** | **2b.** | **2c.** | **3.** |
| SuPr | 0.5 | **0.8** | 0.2 | 0.5 | 0.45 |
| SuPE | 0.25 | 0.25 | 0 | 0.25 | **0.5** |
| PDtP | 0 | 0.04 | 0 | 0.04 | **0.57** |
| PDtG | 0.5 | **1** | **1** | **1** | **1** |
| SIDoS | 0 | 0.88 | 0 | 0.88 | **1** |
| DoNS | **1** | **1** | 0 | **1** | **1** |
| DeTSA | **1** | 0.25 | 0 | 0.25 | 0.25 |
| SaTS | 0.25 | **0.89** | 0.25 | 0.61 | 0.56 |
| NeIB | **1** | **1** | 0 | **1** | 0.5 |
| SaPP | 0 | 0.38 | 0 | 0 | **1** |
| UP | 0.25 | **1** | 0 | **1** | 0.25 |
| **Cumulative AFI per process stage** | 0.43 | 0.68 | 0.13 | 0.59 | 0.64 |

1 : Designing 2a : Slicing software 2b : Printing profile
2c : G-code file 3: Firmware

Table 3: Stage-wise feasibility summary for attack goals

and 0.56 for Stage 3 (firmware). The normalized AFI value for DoPS is 0.43 for Stage 2a and 0.79 for Stage 3. The Cumulative AFI indicates almost equal feasibility at the slicing (2a) and firmware (3) stages, with the print profile being the least feasible stage (2b) for launching an attack.

## 7 Firmware Attack Countermeasures

To fortify 3D printing systems against firmware attacks, a robust, integrated approach encompassing prevention, detection, and mitigation strategies is essential. Primary vectors for firmware attacks include compromises in the supply chain, unauthorized physical access during the operational phase, and intrusions into the printer network and the trusted printer control software. Supply chain security is a comprehensive field that involves safeguards to protect against breaches of trust at any stage. Some of these measures include using encryption and blockchain technology to ensure data integrity and prevent counterfeiting [57]. To overcome the computational constraints of 3D printers, lightweight cryptographic solutions such as Elliptic Curve Cryptography (ECC) and the Elliptic Curve Digital Signature Algorithm (ECDSA) provide suitable encryption and authentication services [58]. To thwart Man-in-the-Middle (MiTM) attacks, secure communication protocols, like TLS, should be implemented to encrypt data exchanges involving the printer. The printer control software, endowed with high-level privileges like firmware installation, must be fortified with robust authentication and authorization measures. To prevent booting from or upgrading to malicious firmware, industry-standard techniques such as cryptographic signing of firmware files, secure boot mechanisms, and the integration of hardware-based security modules like TPM (Trusted Platform Module) or HSM (Hardware Security Module) should be implemented.

A reliable firmware acquisition and subsequent static analysis can help identify malicious code. A hardware-based firmware acquisition method utilizing debugging ports, such as JTAG, effectively bypasses many upper-level deception tactics employed by attackers to evade detection [59]. As a scalable alternative to static analysis, a future direction could involve examining running firmware through cyber-physical fuzzing. The solution would monitor the printer's state in response to smartly generated application-layer probes (G-codes) in a closed loop to promptly expose any malicious behavior within the firmware.

Should an attacker circumvent these preventative measures, additional safeguards can prevent malicious firmware from fulfilling its intended goals. A signature-based anomaly detection solution would be beneficial for detecting malicious firmware behavior. A more comprehensive approach involves a cyber-physical anomaly detection system that analyzes both physical-operational data such as acoustic, electric current, and magnetic fields [53, 60, 61] and digital domain data such as network traffic and application logs. This system can utilize heuristics or machine learning techniques to identify attack signatures and behavioral anomalies. Another forward-looking strategy involves integrating quality control measures into the cybersecurity loop. Since physical processes are inherently imperfect, resulting in low-magnitude deviations, it is crucial to differentiate between benign and harmful deviations. To this end, implementing feasible versions of standard quality control processes, such as real-time micro CT scanning, could enhance the anomaly detection capabilities [62].

These multi-layered strategies will significantly enhance the defense of 3D printing setups against the continually evolving landscape of cyberattacks.

## 8 Conclusion

This study presents a novel approach to understanding and classifying firmware attacks in additive manufacturing. We propose a firmware attack classification tree focused on attack goals rather than attack actions. Additionally, nine attacks on Marlin firmware are demonstrated on the Ultimaker2+ 3D printer. Through a series of destructive and non-destructive tests, including tensile strength and air-quality testing, we confirm the effectiveness of these attacks. To analyze the attacks, we introduce an Attack Feasibility Index (AFI), representing a feasibility score for an attack at a specific stage of the printing process. An analysis of 48 attacks, including existing and proposed ones, confirms that all attack goals could not be achieved by attacking any single stage of the printing process. We observe that firmware is not the optimal stage to launch attacks aimed at sabotaging the printed part. This study will inspire further research into additive manufacturing attacks and guide cybersecurity researchers in developing defense solutions tailored to specific stages of the printing process and their corresponding feasible attacks.

## References

[1] KBV Research, "Share & industry trends analysis report by type, by technology, by sales channel, by end-use, by regional outlook and forecast, 2021–2027," *ReportLinker: Lyon, France*, 2022.

[2] GE Aviation. (2018) New manufacturing milestone: 30,000 additive fuel nozzles. Online Available: https://www.ge.com/additive/stories/new-manufacturing-milestone-30000-additive-fuel-nozzles.

[3] N. Gupta, C. Weber, and S. Newsome, "Additive manufacturing: status and opportunities," *Science and Technology Policy Institute, Washington*, 2012.

[4] S. R. Chhetri, S. Faezi, N. Rashid, and M. A. Al Faruque, "Manufacturing supply chain and product lifecycle security in the era of industry 4.0," *Journal of Hardware and Systems Security*, vol. 2, no. 1, pp. 51–68, 2018.

[5] A. Muhammad, B. Afzal, B. Imran, A. Tanwir, A. H. Akbar, and G. Shah, "onem2m architecture based secure mqtt binding in mbed os," in *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, 2019, pp. 48–56.

[6] A. Ayub, N. Zubair, H. Yoo, W. Jo, and I. Ahmed, "Gadgets of gadgets in industrial control systems: Return oriented programming attacks on plcs," in *2023 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2023, pp. 215–226.

[7] M. Ahsan and M. Ali, "Lsstk: Lightweight solution to preventing stack from buffer overflow vulnerability," in *2023 17th International Conference on Open Source Systems and Technologies (ICOSST)*, 2023, pp. 1–7.

[8] S. A. Qasim, A. Ayub, J. Johnson, and I. Ahmed, "Attacking the iec 61131 logic engine in programmable logic controllers," in *Critical Infrastructure Protection XV: 15th IFIP WG 11.10 International Conference, ICCIP 2021, Virtual Event, March 15–16, 2021, Revised Selected Papers 15*. Springer, 2022, pp. 73–95.

[9] B. Imran, M. Ahsan, A. H. Akbar, and G. A. Shah, "D4gw: Dtls for gateway multiplexed application to secure mqtt(sn)-based pub/sub architecture," *Internet of Things*, vol. 26, p. 101172, 2024.

[10] B. Imran, B. Afzal, A. H. Akbar, M. Ahsan, and G. A. Shah, "Misa: Minimalist implementation of onem2m security architecture for constrained iot devices," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[11] A. Ayub, W. Jo, S. A. Qasim, and I. Ahmed, "How are industrial control systems insecure by design? a deeper insight into real-world programmable logic controllers," *IEEE Security & Privacy*, vol. 21, no. 4, pp. 10–19, 2023.

[12] J. Gatlin, S. Belikovetsky, Y. Elovici, A. Skjellum, J. Lubell, P. Witherell, and M. Yampolskiy, "Encryption is futile: Reconstructing 3d-printed models using the power side-channel," ser. RAID '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 135–147.

[13] M. A. Al Faruque, S. R. Chhetri, A. Canedo, and J. Wan, "Acoustic side-channel attacks on additive manufacturing systems," in *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, 2016, pp. 1–10.

[14] S. Belikovetsky, M. Yampolskiy, J. Toh, J. Gatlin, and Y. Elovici, "dr0wned–cyber-physical attack with additive manufacturing," in *11th USENIX Workshop on Offensive Technologies (WOOT 17)*. Vancouver, BC: USENIX Association, Aug. 2017.

[15] M. H. Rais, M. Ahsan, V. Sharma, R. Barua, R. Prins, and I. Ahmed, "Low-magnitude infill structure manipulation attacks on fff-based 3d printers," in *Critical Infrastructure Protection XVI*. Springer, 2022, pp. 205–232.

[16] N. Gupta, A. Tiwari, S. T. Bukkapatnam, and R. Karri, "Additive manufacturing cyber-physical system: Supply chain cybersecurity and risks," *IEEE Access*, vol. 8, pp. 47 322–47 333, 2020.

[17] M. H. Rais, Y. Li, and I. Ahmed, "Dynamic-thermal and localized filament-kinetic attacks on fused filament fabrication based 3d printing process," *Additive Manufacturing*, p. 102200, 2021.

[18] B. Jovanović, I. Gadjanski, J. Burazer, L. Nikolić, N. Babić, and M. Lečić, "R&d in a fab lab: Examples of paste extrusion method," in *Proceedings of 5th International Conference on Advanced Manufacturing Engineering and Technologies*, V. Majstorovic and Z. Jakovljevic, Eds. Cham: Springer International Publishing, 2017, pp. 461–467.

[19] ISO/ASTM 52900:2021, "Additive manufacturing — General principles — Fundamentals and vocabulary," *ASTM International, West Conshohocken, PA*, 2021. [Online]. Available: https://www.iso.org/standard/74514.html

[20] M. Yampolskiy, W. E. King, J. Gatlin, S. Belikovetsky, A. Brown, A. Skjellum, and Y. Elovici, "Security of additive manufacturing: Attack taxonomy and survey," *Additive Manufacturing*, vol. 21, pp. 431–457, 2018.

[21] M. Yampolskiy, A. Skjellum, M. Kretzschmar, R. A. Overfelt, K. R. Sloan, and A. Yasinsac, "Using 3d printers as weapons," *International Journal of Critical Infrastructure Protection*, vol. 14, pp. 58–71, 2016.

[22] Y. Pan, J. White, D. Schmidt, A. Elhabashy, L. Sturm, J. Camelio, and C. Williams, "Taxonomies for reasoning about cyber-physical attacks in iot-based manufacturing systems," *International Journal of Interactive Multimedia and Artificial Intelligence*, 2017.

[23] P. Mahesh, A. Tiwari, C. Jin, P. R. Kumar, A. N. Reddy, S. T. Bukkapatanam, N. Gupta, and R. Karri, "A survey of cybersecurity of digital manufacturing," *Proceedings of the IEEE*, vol. 109, no. 4, pp. 495–516, 2020.

[24] M. Wu and Y. B. Moon, "Taxonomy of cross-domain attacks on cybermanufacturing system," *Procedia Computer Science*, vol. 114, pp. 367–374, 2017.

[25] C. Song, F. Lin, Z. Ba, K. Ren, C. Zhou, and W. Xu, "My smartphone knows what you print: Exploring smartphone-based side-channel attacks against 3d printers," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 895–907.

[26] M. A. Al Faruque, S. R. Chhetri, A. Canedo, and J. Wan, "Acoustic side-channel attacks on additive manufacturing systems," in *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, 2016, pp. 1–10.

[27] Q. Do, B. Martini, and K.-K. R. Choo, "A data exfiltration and remote exploitation attack on consumer 3d printers," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 10, pp. 2174–2186, 2016.

[28] X. Z. Hang, "Three demos of attacking arduino and reprap 3d printers, code to Keynote at XCon2013 (2013)," https://github.com/secmobi/attack-arduino-and-reprap, 2016.

[29] S. B. Moore, W. B. Glisson, and M. Yampolskiy, "Implications of malicious 3d printer firmware," in *Proceedings of Hawaii Int. Conf.Syst.Sci,2017*, 2017, pp. 1–10.

[30] H. Pearce, K. Yanamandra, N. Gupta, and R. Karri, "Flaw3d: A trojan-based cyber attack on the physical outcomes of additive manufacturing," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 6, pp. 5361–5370, 2022.

[31] Q. Do, B. Martini, and K. R. Choo, "A data exfiltration and remote exploitation attack on consumer 3d printers," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 10, pp. 2174–2186, 2016.

[32] A. Cui, M. Costello, and S. Stolfo, "When firmware modifications attack: A case study of embedded exploitation," in *20th Annual NDSS symposium*, 2013.

[33] M. Yampolskiy, L. Graves, J. Gatlin, A. Skjellum, and M. Yung, "What did you add to my additive manufacturing data?: Steganographic attacks on 3d printing files," in *Proceedings of the 24th International Symposium on Research in Attacks, Intrusions and Defenses*, ser. RAID '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 266–281.

[34] R. Colella, F. P. Chietera, and L. Catarinucci, "Analysis of fdm and dlp 3d-printing technologies to prototype electromagnetic devices for rfid applications," *Sensors*, vol. 21, no. 3, 2021.

[35] Y. Gao, W. Wang, Y. Jin, C. Zhou, W. Xu, and Z. Jin, "Thermotag: A hidden id of 3d printers for fingerprinting and watermarking," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 2805–2820, 2021.

[36] M. H. Rais, M. Ahsan, and I. Ahmed, "Fromepp: Digital forensic readiness framework for material extrusion based 3d printing process," *Forensic Science International: Digital Investigation*, vol. 44, p. 301510, 2023.

[37] K. e. a. Babu, "Fire behavior of 3d-printed polymeric composites," in *Journal of Materials Engineering and Performance*. Springer, 2021, pp. 30:4745–4755.

[38] Q. Zhang, M. Pardo, Y. Rudich, I. Kaplan-Ashiri, J. P. S. Wong, A. Y. Davis, M. S. Black, and R. J. Weber, "Chemical composition and toxicity of particles emitted from a consumer-level 3d printer using various materials," *Environmental Science & Technology*, vol. 53, no. 20, pp. 12 054–12 061, 2019, pMID: 31513393.

[39] S. Pirela, J. Martin, D. Bello, and P. Demokritou, "Nanoparticle exposures from nano-enabled toner-based printing equipment and human health: state of science and future research needs," *Critical reviews in toxicology*, vol. 47, pp. 1–27, 05 2017.

[40] E. Kurkowski, A. V. Stockum, J. Dawson, C. Taylor, T. Schulz, and S. Shenoi, "Manipulation of g-code toolpath files in 3d printers: Attacks and mitigations," in *Critical Infrastructure Protection XVI*. Springer, 2022, pp. 205–232.

[41] T. Hudson, X. Kovah, and C. Kallenberg, "Thunderstrike 2: Sith strike," *Black Hat USA Briefings*, 2015.

[42] C. Kallenberg and R. Wojtczuk, "Speed racer: Exploiting an intel flash protection race condition," *Bromium Labs (January 2015)*, 2015.

[43] D. Ibdah, N. Lachtar, A. A. Elkhail, A. Bacha, and H. Malik, "Dark firmware: a systematic approach to exploring application security risks in the presence of untrusted firmware," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020, pp. 413–426.

[44] L. Garcia, F. Brasser, M. Cintuglu, A. R. Sadeghi, O. Mohammed, and S. A. Zonouz, "Hey, my malware knows physics! attacking plcs with physical model aware rootkit," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, ser. NDSS,2017. Reston, VA, USA: Internet Society, 2017.

[45] T. H. Gabriel Boyd. 3D printing bridging: 6 tips for perfect bridges. https://all3dp.com/2/bridging-3d-printing-tips-tricks-for-perfect-bridges/. Last Updated: Nov 12, 2022.

[46] A. Sinha, B. Swain, A. Behera, P. Mallick, S. K. Samal, H. M. Vishwanatha, and A. Behera, "A review on the processing of aero-turbine blade using 3d print techniques," *Journal of Manufacturing and Materials Processing*, vol. 6, no. 1, 2022.

[47] ASTM, "Standard test methods for flexural properties of unreinforced and reinforced plastics and electrical insulating materials," Modifed on July 24, 2017.

[48] M. Gaskill. (Jan 6, 2023) EPA Proposes to Strengthen Air Quality Standards to Protect the Public from Harmful Effects of Soot. https://www.epa.gov/newsreleases/epa-proposes-strengthen-air-quality-standards-protect-public-harmful-effects-soot.

[49] S. Alyxandra Van, K. Elizabeth, P. Tiffany, T. Curtis, D. Joel, R. Mason, and S. Sujeet, "Attack-defense modeling of material extrusion additive manufacturing systems," in *Critical Infrastructure Protection XVI*. Springer, 2022, pp. 121–153.

[50] S. Y. Nam, D. Kim, and J. Kim, "Enhanced arp: preventing arp poisoning-based man-in-the-middle attacks," *IEEE communications letters*, vol. 14, no. 2, pp. 187–189, 2010.

[51] S. Moore, P. Armstrong, T. McDonald, and M. Yampolskiy, "Vulnerability analysis of desktop 3d printer software," in *2016 Resilience Week (RWS)*. IEEE, 2016, pp. 46–51.

[52] Y. Gao, B. Li, W. Wang, W. Xu, C. Zhou, and Z. Jin, "Watching and safeguarding your 3d printer: Online process monitoring against cyber-physical attacks," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 2, no. 3, Sep. 2018.

[53] C. Bayens, T. Le, L. Garcia, R. Beyah, M. Javanmard, and S. Zonouz, "See no evil, hear no evil, feel no evil, print no evil- malicious fill patterns detection in additive manufacturing," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 1181–1198.

[54] M. H. Rais, Y. Li, and I. Ahmed, "Spatiotemporal g-code modeling for secure fdm-based 3d printing," in *Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems*, 2021, pp. 177–186.

[55] S. E. Zeltmann, N. Gupta, N. G. Tsoutsos, M. Maniatakos, J. Rajendran, and R. Karri, "Manufacturing and security challenges in 3d printing," *JOM*, vol. 68, no. 7, pp. 1872–1881, Jul 2016.

[56] S. Belikovetsky, Y. A. Solewicz, M. Yampolskiy, J. Toh, and Y. Elovici, "Digital audio signature for 3d printing integrity," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1127–1141, 2019.

[57] S. Zeadally and J. Bou abdo, "Blockchain: Trends and future opportunities," *Internet Technology Letters*, vol. 2, 09 2019.

[58] B. Hammi, S. Zeadally, and J. Nebhen, "Security threats, countermeasures, and challenges of digital supply chains," *ACM Comput. Surv.*, vol. 55, no. 14s, jul 2023. [Online]. Available: https://doi.org/10.1145/3588999

[59] M. H. Rais, R. A. Awad, J. Lopez, and I. Ahmed, "Jtag-based plc memory acquisition framework for industrial control systems," *Forensic Science International: Digital Investigation*, vol. 37, p. 301196, 2021.

[60] J. Gatlin, S. Belikovetsky, S. B. Moore, Y. Solewicz, Y. Elovici, and M. Yampolskiy, "Detecting sabotage attacks in additive manufacturing using actuator power signatures," *IEEE Access*, vol. 7, pp. 133 421–133 432, 2019.

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

[62] M. Ahsan, M. H. Rais, and I. Ahmed, "Sok: Side channel monitoring for additive manufacturing - bridging cybersecurity and quality assurance communities," in *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, 2023, pp. 1160–1178.

## A Appendix: Attacks Description

Table 4 gives a generic description of all the analyzed attacks.

| Sr. # | Attack Name | Attack Action |
|---|---|---|
| 1 | Printer info | Exfiltrating printer information e.g. manufacturer, model, firmware version, etc. |
| 2 | Design SW info | Exfiltrating CAD software information e.g. name, version, etc. |
| 3 | Slicer/Control info | Exfiltrating slicing printer control software information e.g. name, version, etc. |
| 4 | OG info | Stealing printed object geometry information (IP Theft) |
| 5 | Print profile info | Extracting printing profile (thermal, infill pattern, density, etc,) to facilitate |
| 6 | Network device | Using compromised printer to extract networked devices information |
| 7 | Process artefacts | Using AM process artefacts as information carrier |
| 8 | Facility info | Printing facility information e.g. stealing environment temperature, cameras, etc. |
| 9 | PYOG | Print your own grave: Beak the printing glass |
| 10 | Breaking limits | Making printer go beyond limits to cause damage to the end-stops/limit switches |
| 11 | Nz impair | Hitting the nozzle to the print bed to physically damage the nozzle orifice |
| 12 | Extruder fracture | Hitting the extruder assembly against the printer walls to physically fracture it |
| 13 | Nz burning | Heating the nozzle for a longer period with the cooling fan turned off |
| 14 | OG scaling | Scaling up or down the print object outer geometry |
| 15 | OG thermal | Deforming the object geometry through thermodynamic manipulation by reducing the fan speed or changing its state. |
| 16 | Incurable | Impersonating low quality bridging defect (Section 5.3) |
| 17 | Warping | Adding warping defects to the print geometry by changing thermal parameters |
| 18 | FK thermal | Lowering nozzle temperature resulting in cold filament extrusion |
| 19 | Trajectory unsync | Unsynchronized nozzle trajectory for x,y,e axes |
| 20 | PS profile | Manipulating the trapezoidal speed profile to cause excessive nozzle jerks |
| 21 | PS unsync | Unsynchronized x,y extruder speed during printing |
| 22 | FK reduction | Decreasing feed-rate to cause material underflow |
| 23 | Clogging | Partially clog the printer nozzle resulting in material underflow |
| 24 | MAC/ARP corruption | Denying printer access by manipulating mac table or through ARP poisoning |
| 25 | Vertex relocation | Relocating one or more selected vertices |
| 26 | OF scaling | Scaling up/down print object specific feature |
| 27 | Fitment | Axial misalignment of the print feature to cause fitment issues for intended assembly |
| 28 | NT manipulation | Traffic manipulation to breach network integrity |
| 29 | IF line spacing | Changing infill-lines spacing to reduce build part strength |
| 30 | IF vertex spacing | Changing infill vertices spacing to reduce build part strength |
| 31 | FS manipulation | Filament-state manipulation to evade nozzle kinetic detectors |
| 32 | FK cavity | Cavity through filament-kinetics w/o modifying toolpath |
| 33 | FK density | Localized density variation by filament status/speed change |
| 34 | IF pattern | Changing the Infill pattern for example from honeycomb to linear etc. |
| 35 | IF density | Changing the Infill density (1% or more) |
| 36 | PS cavity | Modifying printing speed for localized zones |
| 37 | IF exclusion | Excluding the infill pattern in the print geometry |
| 38 | % IF | Changing % fill for the infill pattern e.g. from 50% to 25% and making it more sparse |
| 39 | GF change | Replacing the printing instructions (G-code) file |
| 40 | Dyn. thermal / bonding | Manipulating interlayer bonding by changing thermal properties at localized zones |
| 41 | PS local | Manipulating the printing speed at localized regions |
| 42 | OF insert/remove | Adding or removing a geometric feature in the print geometry |
| 43 | LT local | Localized changes to layer thickness by manipulating z-profile |
| 44 | GC sequence | Localized modification in the toolpath sequence e.g., following a different printing path |
| 45 | Anisotropy | Changing print direction to vary anisotropic properties of the print object |
| 46 | GC manipulation | Insertion, removal, or modification of the printing instructions |
| 47 | Erosion | Causing filament erosion based density attack |
| 48 | PF air quality | Microparticles and VOC flooding to degrade air quality of the printing facility |

Table 4: Generic description of the studied attacks

## B Appendix: Algorithms for Firmware Attacks

**Algorithm 1** Printed Object Surveillance Attack

```
Output: Object sketch file theft
Phase-1: Sketch Compilation
On restarts: $*\text{eeprom}_{\text{Atk}_{\text{end}}} \rightarrow$ spyFile
if G-code == G0 or G1 then
    if not spyFile then
        if L.Change() && $Z_{\text{dst}} \geq (E_{\text{no}} * Z_{\text{a}} + 1)$ then
            Shell ← Find_Shell()
            $*\text{eeprom}_{\text{loc}} \leftarrow$ L.Header; loc++
            $*\text{eeprom}_{\text{loc}} \leftarrow Z_{\text{dst}}$; loc++
            for P in Shell do
                $*\text{eeprom}_{\text{loc}} \leftarrow P_x$; loc++
                $*\text{eeprom}_{\text{loc}} \leftarrow P_y$; loc++
            end for
        else
            if $Z_{\text{dst}} == Z_{\text{current}}$ then
                Queue ← Queue ∪ $\text{P}_{x,y}$
            else
                if printingDone() then
                    Queue.reset()
                    spyFile = 1
                    $*\text{eeprom}_{\text{Atk}_{\text{end}-1}} \leftarrow (loc - loc_{\text{o}})$
                    $*\text{eeprom}_{\text{Atk}_{\text{end}}} \leftarrow 0x01$
                    $\text{Reset}_{\text{Queue, loc}}$
                end if
            end if
        end if
    end if
end if
Continue_execution
Phase-2: File transfer
if SDinserted && SDstateChange then
    if spyFile then
        if SDauthenticate() then
            SD.openFile("spidy.txt", 'w')
            for i = 0 to $*\text{eeprom}_{\text{Atk}_{\text{end}-1}}$ do
                SD.write($*\text{eeprom}_{(loc_{\text{o}}+i)}$)
            end for
            SD.closeFile()
            spyFile = 0
            $*\text{eeprom}_{\text{Atk}_{\text{end}-1}} \leftarrow 0x00$
            $*\text{eeprom}_{\text{Atk}_{\text{end}}} \leftarrow 0x00$
        end if
    end if
end if
```

**Algorithm 2** Print Your Own Grave: Break the Glass

```
Output: Breaking the printing glass
Trigger: An unused G-code G98
Preheat the printing bed and nozzle
for layer = 1 to n do                          ▷ n is the desired number of layers
    $x \leftarrow 112.5 + \text{osc} \times 0.1$
    $y \leftarrow 112.5 + \text{osc} \times 0.1$
    $\text{osc} \leftarrow -\text{osc}$
    if layer $> 8$ then
        line-count ← small-square               ▷ Destruction tool feature 1
    else
        line-count ← big-square                 ▷ Destruction tool feature 2
    end if
    for line = 1 to m do                        ▷ m is the number of lines
        if line $< 4$ then
            speed ← slow
        else if layer $> 4$ then
            speed ← moderate
        else
            speed ← fast
        end if
        $x \leftarrow x + \text{dir} \times \text{lenx}$
        $y \leftarrow y + \text{dir} \times \text{leny}$
        $e \leftarrow e + \text{dir} \times \text{lene}$
        Move to $(x, y, e)$
        lenx, leny ← lenx, leny + 0.8
        lene ← 0.058 × lenx
    end for
end for
while nozzle and printing bed cool down do
    wait!
end while
Grip(printed-tool): Nozzle jams in cavity and holds the tool
Unlock(retaining-clips)
Manipulate $y, z$ position variables
Move nozzle tip beyond and below glass sheet
Guide the glass out through the walls and dispose
```

**Algorithm 3** Attack to Simulate Bridging Errors Over X-Axis

```
Output: Poor bridging performance
Context: Attack resides within Move instruction code region
G-code instruction: Move from $A$ to $B$
if $B_z < \text{Layer}_{\text{width}}$ then
    Initialize layer-number
else if $B_z > A_z$ then
    Increment layer-number
    Copy $\text{LMAP}_{\text{current}}$ to $\text{LMAP}_{\text{prev}}$   # Layer Map
else if $\Delta e > 0 \wedge \Delta x \neq 0$ then
    direction $\leftarrow (B_x > A_x)\ ?\ +1 : -1$
    xvar $\leftarrow \text{round}(A_x, 20\text{ mm})$
    yvar $\leftarrow \text{round}(A_y, 20\text{ mm})$
    while xvar $< B_x$ do
        if xvar within Attack-Zone then
            $(i, j) \leftarrow \text{LMAP}_{\text{ref}}$ index for $(xvar, yvar)$
            if $\text{LMAP}_{\text{prev}}[i, j] == 0$ then
                attack-the-command $\leftarrow$ true
                $\text{LMAP}_{\text{current}}[i, j] = 1$
            end if
            xvar $\leftarrow$ xvar $+ 20$ mm
        end if
    end while
    if attack-the-command then
        Modify extruder settings:
            $T \leftarrow T + 5^{\circ}C$        ▷ Increase temperature by $5^{o}$C
            $F \leftarrow 0.5 \times F$          ▷ Reduce feedrate to 50%
            $S \leftarrow 0.5 \times S$          ▷ Reduce fan speed to 50%
            $L \leftarrow 1.25 \times L$         ▷ Increase extrusion length by 25%
        Execute the move command
        Revert modifications to $T, F, S, L$
        attack-the-command $\leftarrow$ false
    end if
end if
```

**Algorithm 4** Object Feature Scaling Attack

```
Output: Enlarged geometry over x and y axes
Initialize new object
G-code instruction rx: Move from $A$ to $B$
if $B_z > A_z$ then
    while Queue-size $\geq 3$ do
        Update Tail position in queue
        initialize Polygon-found $\leftarrow$ false
        while Head not reached do
            Traverse the queue
            if Tail coordinates found then
                Polygon-found $\leftarrow$ true
                break
            end if
        end while
        if Polygon-found then
            break
        end if
        Decrement Queue-size
    end while
    if Polygon-found then
        $P_{\text{tail}'} \leftarrow$ Find position outside polygon adjacent to $P_{\text{tail}'}$
        Move to $P_{\text{tail}'}$
        for $P_i \in$ Tail to Head do
            $P'_i \leftarrow$ Find corresponding position for $P_i$
            $P_{i'_e} \leftarrow P_{i_e}$
            Move to $P'_i$
        end for
        Adjust $P_{\text{Tail}_e}$ for extra filament used
        Attack accomplished for the current layer
        Reset Queue for the next layer
    end if
else if $B_z = A_z$ then
    Add $B$ at Tail; Update Head and Tail positions
else
    Reset Queue
end if
```

**Algorithm 5** Misalignment Attack

```
Output: Misaligning an axial slot by $\theta^{o}$ to cause fitment error
Initialize new object
if Polygon-found then                        ▷ logic defined in Algorithm 4
    temporalCountingStart = True             ▷ increase temporalDiff
    if (layer-change) then
        if temporalDiff > minGap then        ▷ Internal feature found
            for eachG-code $\in$ polygon do
                Either x $\leftarrow$ x + (layerHeight / tan(90-θ))
                Or y $\leftarrow$ y + (layerHeight / tan(90-θ))
                Or x,y $\leftarrow$ x,y + (layerHeight / tan(90-θ))
            end for
        else                                 ▷ No internal feature found
            misalignCompleteObject OR skipAttack
            execute_buffered_G-codes
        end if
        Reset Queue for the next layer
    end if
end if
```

**Algorithm 6** Internal Cavity Attack

```
Output: A cavity inside the object
Procedure:
if layerCount == 0 then    ▷ To find total commands in a layer
    cmdPerLayer ← cmdPerLayer + 1
end if
if $Z_{\text{new}} > Z_{\text{old}} + \text{minLayerWidth}$ then
    layerChange ← true
    targetCmdNo ← $\frac{\text{cmdPerLayer}}{2} - 2$
end if
if $30 \leq M73.\text{value} \leq 70$ then    ▷ Printing internal layers
    attackStatus ← true
    layerChange ← false
end if
if attackStatus == True then
    currentCmd ← currentCmd + 1
    if currentCmd = targetCmdNo ± 2 then
        $C_1, C_2, C_3$ ← split_symmetric(currentCmd)
        $C_2$ ← mute_extruder_part($C_2$)
        skip G-code(currentCmd)
        execute G-code($C_1$)
        retract_filament(4 mm)
        execute G-code($C_2$)
        advance_filament(4 mm)
        execute G-code($C_3$)
    end if
end if
```

**Algorithm 7** Object Density Variation Attack

```
Output: Low-density zones in the internal layers
Procedure:
if layerCount == 0 then    ▷ To find total commands in a layer
    cmdPerLayer ← cmdPerLayer + 1
end if
if $Z_{\text{new}} > Z_{\text{old}} + \text{minLayerWidth}$ then
    layerChange ← true
    targetCmdNo ← $\frac{\text{cmdPerLayer}}{2} - 2$
end if
if $30 \leq M73.\text{value} \leq 70$ then    ▷ Printing internal layers
    attackStatus ← true
    layerChange ← false
end if
if attackStatus then
    currentCmd ← currentCmd + 1
    if currentCmd = targetCmdNo ± 2 then
        modifiedCmd = mute_extruder_part(currentCmd)hg/
        skip G-code(currentCmd)
        execute modifiedCmd
    end if
end if
```

**Algorithm 8** Filament Erosion Attack

```
Output: Reduced filament quantity
Procedure:
Method-1
if printingInitiates==True then    ▷ Suitable at the start of printing
    if G-code== heatNozzle($T_n$) then    ▷ preheating command
        bufferG-code()
        setExtrudeMinTemp($0^oC$) ▷ To bypass the safety check
        moveExtruder(30)    ▷
        executeHeatNozzle($T_n$)
        restoreExtruderMinTemperature
        printingInitiates = False
    end if
end if
Method-2
if printingStatus==True then    ▷ It is repeated throughout printing
    estimateAttackZone    ▷ Based on object & bowden tube
    if attackZone==True then    ▷ x times in a layer
        eliminateMaxChecks(speed,acceleration, jerk)
        retreatFilament(5)    ▷ To avoid any spilling
        for i ∈ oscCount do    ▷ high-jerk moves, 20-50
            peakJerkMoves(±4)    ▷ to-and-fro at peak settings
        end for
        advanceFilament(5)
    end if
end if
```

**Algorithm 9** Printing Facility Air Quality Attack

```
Output: Degraded air quality of the printing facility
Procedure:
if idleDuration > inactivity_threshold then    ▷ Establish idle status by the absence of temperature and movement G-codes
    if nozzleTemperature < 150 then    ▷ suitable for cold extrusion
        initiateColdExtrudeBursts()    ▷ using Algorithm 8
        preheatNozzle(180)    ▷ using M109 G-code
        retractFilament(4)    ▷ to avoid drops on the bed
        disableHeaterFeedback()
        switchOnHeater()    ▷ may achieve $T_n$ up to $350^oC$
        hold_and_wait()    ▷ 1-5 mins
        switchOfHeater()
        enableHeaterFeedback()
    end if
    resetIdleDuration()
end if
```