Secure and Remote 3D Printing

By: Tiffanie Petersen, Isaiah Thomas, Carl Mann, & Nick Cottrell

General Information:

Meeting Dates:

- 3/21/22

Client

Mike C Newton

Faculty Sponsor

Dr. Sid Bhattacharyya

Milestone 5

Task	Tiffanie	Carl	Isaiah	Nick
Https overhaul				**100%**
Begin MitM of 3D printer		**70%**	**30%**	
Create the poster for the project	**90%**		**10%**	
Compile (1st round) evaluation results	**100%**			
Create e-book page	**5%**		**95%**	

HTTPS Demo

E-book Page

Goals and motivation

Security measures taken

Pen testing and evaluations

[Capstone Category]

Project Name **Secure & Remote 3D Printing**
Team Lead: Tiffanie Petersen
Team Member(s): Isaiah Thomas, Nick Cotrell, Carl Mann
Faculty Advisor(s): Dr. Siddhartha Bhattacharyya, Dept. of Computer Engineering and Sciences, Florida Institute of Technology

****do not change font size or text color above this message/delete this before completion or put in a category. The category will be put in by Staff after submission ****

For many industries the importance of 3D printing in manufacturing processes has grown immensely. The initial concern which sparked the development of this project was that 3D printers have been historically attacked in order to cause them to stray from their original designs and inject various modifications. This could lead to wasting materials, the destruction of important projects, or the printing of undesired modifications which may change the functionality of a model. Another concern was that in most cases 3D printing must be initiated on site on a machine with a wired connection. The goal of this project was to negate these concerns by providing a client with a secure and remote method of 3D printing. This was accomplished in two stages.

For the first stage we were tasked to design and develop a web application to schedule, approve/decline, and monitor the process of an Ender-3 3D printer. After investigating several web frameworks with pre-existing and tested security measures we decided to utilize Django. Over the course of the first semester we developed a site with a focus on ease of use while also implementing security measures to protect users and their uploaded models. Users can use this web application to send 3D project files to a 3D printer remotely. This means that users are able to start their projects while being off property, thus being more convenient than conventional 3D printing methods. The remote connection is structured in a way in which a user can easily leave and return to the application as desired to check on the status of a printing job. By providing users with an estimate for time of completion they will waste less time checking on progress and be able to plan picking up their model accordingly. Security measures implemented to protect remote users and the client include HTTPS, CSRF tokens, uploaded file screening, authorized user whitelist, administrative control of prints, and non-exposed local handling of files between the printer and the site. Administrators of the site have several powerful capabilities at their disposal. One of these is the ability to view uploaded gcode models in an embedded viewer to ensure the model meets whatever guidelines are set by the client. Administrators are also able to perform operations such as adding files, deleting files, starting prints, pausing prints, canceling prints, and homing the printer. During the end phase of website development we shifted focus to implementing a channel for communication with a printer. To accomplish this we used Octoprint to handle the backend components of the application such as keeping track of the printing process. Octoprint is vital because it is capable of keeping track of the queue of projects running while also allowing administrators to collect and swap projects by using simple REST API calls. Communication between the website, octoprint, and the printer is facilitated over a raspberry pi. On this pi, all applications are deployed using docker allowing for easy and effective deployment.

With a functional and secure web application deployed the final stage of development was to investigate potential vulnerabilities and security flaws of the platform. In order to ensure each print job is completed as expected we worked hard to prevent tampering during the staging process and execution of gcode files. One way in which we attempted to prevent tampering was to research potential ways to proxy usb interfaces using the embedded tool the GreatFET. This would allow us to listen to commands currently being executed by the printer and cross reference this with the expected commands in the given gcode file. Establishing a proxy between the printer and octoprint has yet to be established however there has been much success intercepting other devices such as mice and keyboards. A restricted beta has been conducted and provided us with valuable reviews of the platform which we plan on using to improve upon its implementation further. After two semesters of development we believe we provide both users and a client with a streamlined and secure approach to remote 3D printing.

MitM Progress

```
┌──(kali㉿dell-station-03)-[~/Facedancer]
└─$ lsusb
Bus 002 Device 002: ID 0e8d:7612 MediaTek Inc. MT7612U 802.11a/b/g/n/ac Wireless Ada
pter
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 005: ID 093a:2533 Pixart Imaging, Inc. Gaming Mouse
Bus 001 Device 004: ID 187c:0550 Alienware Corporation LED controller
Bus 001 Device 023: ID 2222:0031 MacAlly Macally SLIMKEYC USB Keyboard
Bus 001 Device 022: ID 1d50:60e6 OpenMoko, Inc. replacement for GoodFET/FaceDancer -
 GreatFet
Bus 001 Device 007: ID 0cf3:e009 Qualcomm Atheros Communications
Bus 001 Device 006: ID 04d9:1836 Holtek Semiconductor, Inc. Mechanical Keyboard
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Device to be proxied

MitM Cont.

```
┌──(kali㉿dell-station-03)-[~/Facedancer]
└─$ sudo -E python3 facedancer-usbproxy.py -v 2222 -p 0031
Using GreatDancer backend.
GreatDancer initialized
[17:06:13] <, standard request to device (GET_DESCRIPTOR: value=DEVICE descriptor (index=0×00), index=0, length=64)
[17:06:13] <: b'\x12\x01\x10\x01\x00\x00\x00\x08""1\x00\x00\x01\x01\x02\x00\x01'
-- Patched device descriptor. --
[17:06:14] <, standard request to device (GET_DESCRIPTOR: value=DEVICE descriptor (index=0×00), index=0, length=18)
[17:06:14] <: b'\x12\x01\x10\x01\x00\x00\x00\x08""1\x00\x00\x01\x01\x02\x00\x01'
-- Patched device descriptor. --
[17:06:14] <, standard request to device (GET_DESCRIPTOR: value=CONFIGURATION descriptor (index=0×00), index=0, length=9)
[17:06:14] <: b'\t\x02;\x00\x02\x01\x00\xa02'
[17:06:14] <, standard request to device (GET_DESCRIPTOR: value=CONFIGURATION descriptor (index=0×00), index=0, length=59)
[17:06:14] <: b'\t\x02;\x00\x02\x01\x00\xa02\t\x04\x00\x00\x01\x03\x01\x01\x00\t!\x10\x01\x00\x01"A\x00\x07\x05\x81\x03\x08\x00\n\t\x04\x01\x00\x01\x03\x00\x00\x00\t!\x10\x01\x00\x01"\xb8\x00\x07\x05\x82\x03\x08\x00\n'
-- Storing configuration <USBConfiguration index=1 num_interfaces=2 attributes=0×A0 max_power=100mA> --
[17:06:14] <, standard request to device (GET_DESCRIPTOR: value=STRING descriptor (index=0×00), index=0, length=255)
[17:06:14] <: ₨
[17:06:14] <, standard request to device (GET_DESCRIPTOR: value=STRING descriptor (index=0×02), index=409, length=255)
[17:06:14] <: Macally SLIMKEYC USB Keyboard
[17:06:14] <, standard request to device (GET_DESCRIPTOR: value=STRING descriptor (index=0×01), index=409, length=255)
[17:06:14] <: Macally Peripherals
[17:06:14] >, standard request to device (SET_CONFIGURATION: value=1, index=0, length=0)
-- Applying configuration <USBConfiguration index=1 num_interfaces=2 attributes=0×A0 max_power=100mA> --
[17:06:14] >, class request to interface (class request 10: value=0, index=0, length=0)
[17:06:14] <, standard request to interface (GET_DESCRIPTOR: value=REPORT descriptor (index=0×00), index=0, length=65)
[17:06:14] <: b'\x05\x01\t\x06\xa1\x01\x05\x07\x19\xe0)\xe7\x15\x00%\x01u\x01\x95\x08\x81\x02\x95\x01u\x08\x81\x01\x95\x03u\x01\x05\x08\x19\x01)\x03\x91\x02\x95\x05u\x01\x91\x01\x95\x06u\x08\x15\x008\xff\x00\x05\x07\x19\x00*\xff\x00\x81\x00\xc0'
[17:06:14] >, class request to interface (class request 9: value=200, index=0, length=1)
[17:06:14] >: b'\x00'
[17:06:14] >, class request to interface (class request 10: value=0, index=1, length=0)
[17:06:14] <, standard request to interface (GET_DESCRIPTOR: value=REPORT descriptor (index=0×00), index=1, length=184)
[17:06:14] <: b"\x06\x01\x00\t\x80\xa1\x01\x85\x02%\x01\x15\x00u\x01\n\x81\x00\n\x82\x00\n\x83\x00\x95\x03\x81\x06\x95\x05\x81\x01\xc0\x06\x0c\x00\t\x01\xa1\x01\x85\x03%\x01\x15\x00u\x01\n\xb5\x00\n\xb6\x00\no\x00\np\x00\n\xe2\x00\n\xa2\x00\n\xe9\x00\n\xea\x00\x95\x08\x81\x02\n\x83\x01\n\x94\x01\n\x86\x01\n\x88\x01\n\x8a\x01\n\x92\x01\n\xb7\x00\n\xcd\x00\x95\x08\x81\x02\n!\x02\n#\x02\n$\x02\n%\x02\n6\x02\n'\x02\n*\x02\n\xb8\x00\x95\x08\x81\x02\xc0\x06\x00\xff\t\x01\xa1\x01\x85\x06\x15\x00%\xff\x1e\x01\x00*\x0f\x04u\x08\x96\x0f\x04\xb1\x02\xc0\x06\x00\xff\t\x01\xa1\x01\x85\x05\x15\x00%\xff\x19\x01)\x05u\x08\x95\x05\xb1\x02\xc0"
```

Setting vendor and product ids

Intercepted HID data

```
[17:06:14] >: b'\x01'
[17:06:22] 1<: b'\x00\x00\x0b\x00\x00\x00\x00\x00'
[17:06:22] 1<: b'\x00\x00\x00\x00\x00\x00\x00\x00'
[17:06:22] 1<: b'\x00\x00\x08\x00\x00\x00\x00\x00'
[17:06:22] 1<: b'\x00\x00\x00\x00\x00\x00\x00\x00'
[17:06:22] 1<: b'\x00\x00\x0f\x00\x00\x00\x00\x00'
[17:06:23] 1<: b'\x00\x00\x00\x00\x00\x00\x00\x00'
[17:06:23] 1<: b'\x00\x00\x0f\x00\x00\x00\x00\x00'
[17:06:23] 1<: b'\x00\x00\x00\x00\x00\x00\x00\x00'
[17:06:23] 1<: b'\x00\x00\x12\x00\x00\x00\x00\x00'
[17:06:23] 1<: b'\x00\x00\x00\x00\x00\x00\x00\x00'
[17:06:23] 1<: b'\x00\x00,\x00\x00\x00\x00\x00\x00'
[17:06:23] 1<: b'\x00\x00\x00\x00\x00\x00\x00\x00'
[17:06:23] 1<: b'\x00\x00\x0c\x00\x00\x00\x00\x00'
[17:06:24] 1<: b'\x00\x00\x00\x00\x00\x00\x00\x00'
[17:06:24] 1<: b'\x00\x00\x16\x00\x00\x00\x00\x00'
```

Facedancer listens in on USB device interfacing with host

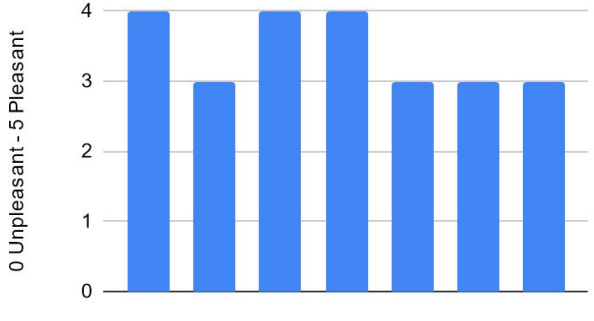
Evaluation Results

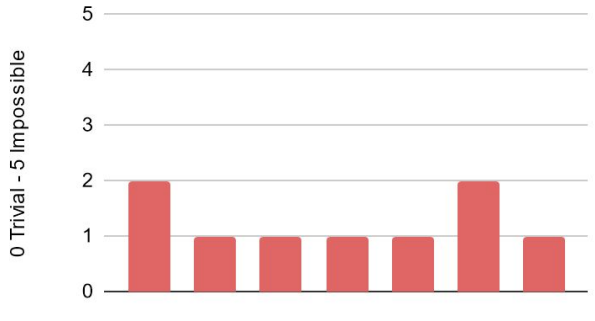
Appearance

- The style is slightly dated and could be cleaner
- Make the shortcuts on the bottom of the screen larger
- Better use of empty space, more style to navbar
- The title doesn't change and needs to be bigger
- The hello world text should be replaced
- Overall graphical/image improvements

User Interaction Feedback

- Provide users with feedback when their print has started
- See progress of uploads in a page
- Multiple uploads
- A robots.txt and sitemap.xml





Evaluation Results

Admin Experience

- More information about file uploads
- Might be helpful to have an undo button in case a file gets deleted on accident
- Button to contact the requester for more info related to the build request
- Blocking an account
- Maybe a way to ban a user. You'd probably just want to delete any banned user's files instead of letting them know they are banned since they could just create a new account to get around the ban.
- Users require admin approval upon account creation before they can upload files. You could easily write a script to spam that webpage and flood it with data from /dev/random. Email validation before being able to upload is also useful.

Additional Feedback

- Minimalistic design with effective functionality serves well for FIT websites

NORTHROP GRUMMAN
ENGINEERING & SCIENCE
STUDENT DESIGN SHOWCASE
FLORIDA TECH

Secure & Remote 3D Printing

Nick Contrell, Carl Mann, Tiffanie Petersen & Isaiah Thomas
Faculty Advisor(s):Dr. Siddhartha Bhattacharyya, Dept. of Computer Engineering and Sciences, Florida Institute of Technology

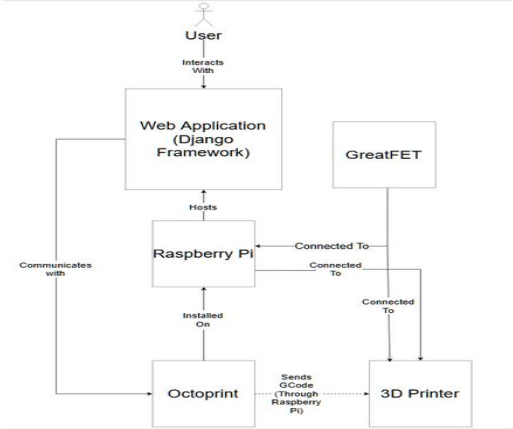
Motivation

- Currently there are exploits for the 3D printer which causes the printer to stray from the original design to create injected modifications
- 3D printers require hands on activity which many users would like to mitigate
- Adding a remote way to monitor the printer would allow administrators to multitask

Goal

- Develop a web application to remotely print an uploaded 3D model
- Have a secure line of communication form user to webserver to printer
- Allow administrators to control a queue of print requests and provide them with the tools necessary to moderate which files should be printed

Design



Features

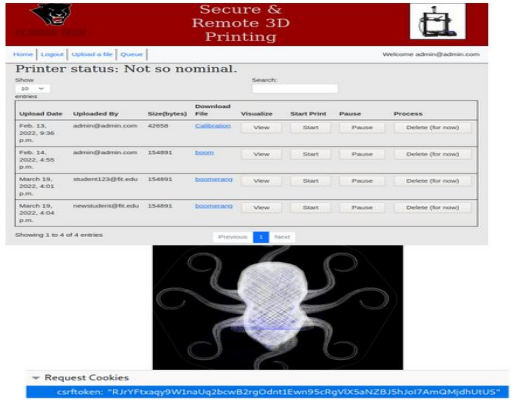
- Website features
  - Users may create an account, login, logout, and upload files to be screened by an administrator
  - Queue view for administrators to monitor, view, and print models
  - Interfaces with Octoprint via REST API calls to pull relevant data and control printer operations

Security Features

- By design
  - Website, octoprint, and file server are all hosted on a raspberry pi with docker
    - Containers prevent an attacker from listening to any internal communications
- Additional measures
  - Encrypted communication channels include HTTPS between the user and the website as well as a direct USB to serial connection between the pi and the printer
  - A Django CSRF Token is used by the server to provide a user with a unique connection specific value to be included in the HTTP requests
  - Extensive file checks prevent users from uploading malicious code and files are stored outside of the projects scope
  - Users are required to create an account in order to upload files

Evaluation

- Recommended features
  - preview 3D projects (implemented)
  - Improve upon the websites appearance
  - Provide users with more feedback relating to their requested prints
- Issues
  - Cannot establish a MiTM connection to the 3D printer reliably, limiting our ability to fuzz traffic



Conclusion

- Users can print remotely while knowing that their projects will come out as expected
- Administrators may view and approve files that have been scanned to ensure they are not malicious
- The web application tracks each of the user uploads and places them in a queue

Future Work

- Provide users feedback when their project has started printing
- Establish a connection to the printer and fuzz gcode input in order to ensure files passed in do not exploit unforseen edge cases
- Add a contact page to reach out to an admin
- Setup an email server to notify users of print job progress

Acknowledgements

- Dr. Chan and peers for provided feedback

Milestone 6

Task	Tiffanie	Carl	Isaiah	Nick
1. Test/demo of the entire system	25%	25%	25%	25%
2. Evaluation results	25%	25%	25%	25%
3. Create user/developer manual	25%	25%	25%	25%
4. Create demo video	25%	25%	25%	25%
5. Finish MitM of printer	25%	25%	25%	25%

Any Questions?

Thank You!