Embedded Systems Security VIP Fall'23 Final Paper: CSAW Team A

Nashad Mohamed
Georgia Institute of Technology
nmohamed9@gatech.edu

Mahta Tavafoghi
Georgia Institute of Technology
mtavafoghi3@gatech.edu

Kayla Kirnon
Georgia Institute of Technology
kkirnon3@gatech.edu

Lindsay Estrella
Georgia Institute of Technology
lestrella7@gatech.edu

***Abstract*—This paper outlines our methodical approach to understanding and addressing the challenges in this year's CSAW competition. We detail our processes for vulnerability analysis, process, and execution. This paper offers insights into our approaches to the challenges we completed and the others that stumped us. The paper also reflects any changes or new approaches done after receiving a partial answer key to the challenges,**

I. Introduction

This year's 2023 CSAW competition focused on side channel attacks (SCA) on cyber-physical systems (CPS). CPS is a combination of physical and computational elements that can be found in use today in many industries. Our initial efforts for this class included general research on the different types of side-channel attacks in order to adequately prepare to complete the qualification paper for CSAW, which was due earlier on in September. This being said, during this research we came to realize that as the industries we would've mentioned prior are growing and becoming more data-heavy, and as a result of this, it is of the utmost importance that we understand the possible effects of these attacks and how they can be mitigated. As stated before were tasked with becoming familiar with the different SCAs and their mitigations. Given our newly acquired knowledge, we exploited these vulnerabilities in order to capture flags. In total, we were given 6 challenges over the course of 3 weeks and rushed to complete as many as we could ahead of the in-person finals in New York City.

II. Electromagnetic Attacks

Electromagnetic side channel attacks use measurements of unintentional electromagnetic radiation from target devices to then carry out signal analysis. This allows the attacker to gain information about the data handling and overall operations of the computing device. This type of attack is becoming increasingly powerful [14] due to the fact that in a noise-induced environment, and from long distances away, the attack can still be carried out successfully.

A. Execution of Electromagnetic Attacks

This type of attack is executed when the attacker is able to measure the electromagnetic radiation being emitted from the target device and then uses this information to perform a signal analysis on it. There are two main ways to study

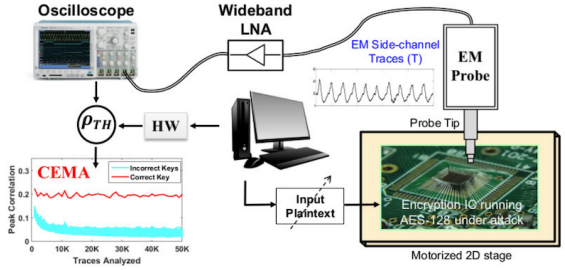

Fig. 1. An example setup for an electromagnetic side-channel attack showing the components present and how they interact while the attack is being carried out [13]

Electromagnetic Radiation: Simple EM Analysis (SEMA) and Differential EM Analysis (DEMA). The data received from these analyses can help the attacker pinpoint certain characteristics or components of the target device that are of interest. This pattern of receiving data allows there to be minimal physical impact on the device being targeted, and this is one of the reasons why this form of attack is so promising to attackers.

An example of an electromagnetic attack was carried out on the Apple CoreCrypto. The researchers were able to successfully extract a 128-bit secret key from the device and used between 5-30 million traces to do so. [10]

B. Mitigation Techniques

The mitigation techniques for electromagnetic attacks discussed in this paper make electromagnetic attacks harder to carry out– they don't completely prevent hackers from carrying them out. There are two overarching groupings of mitigation techniques for electromagnetic attacks: Signal Strength Reduction and Signal Information Reduction.

1) Signal Strength Reduction: Signal strength reduction involves circuit redesign and physically secured zones [13]. Circuit redesign is done to reduce the quantity of unintentional electromagnetic radiation, so there will be less accessible information to begin with. Physically secured zones will reduce the strength of the available signals, which is usually done by introducing various strengths of noise.

2) Signal Information Reduction: Signal information reduction involves randomization or refreshing of keys [5]. This will reduce the effectiveness of attacks on the available signals. An example of one such technique is masking. This is executed by combining the input value with a random value in an attempt to disguise the original input. This technique is effective if differential EM analysis (DEMA) is going to be used.

III. Timing Attacks

A. How attack is carried out

Timing attacks are another variation of a side-channel attack where the time taken for a cryptographic system or software to execute a specific operation is exploited. These attacks are reliant on the fact that various input data can result in observable differences in the timing of the computational operations. So, an attacker accurately measures the exact time a cryptosystem operation takes to reveal information on what the inputs are [1].

An example of a timing attack that was uncovered was the vulnerability known as Meltdown and Spectre, which affect IBM processors, ARM-based processors, and Intel x86 microprocessors. One step of Meltdown was vulnerable to a timing side-channel attack due to the design of the processor, providing attackers the ability to read all memory, which contained sensitive information [1]. Spectre outlined the vulnerabilities within the microprocessors that performed branch prediction. This is when the circuit attempts to guess the direction of the execution sequence, so misbranching can occur, leaving the CPU vulnerable and allowing an attacker to access kernel memory which includes passwords, encryption keys, emails, etc [4]. When these processes are timed, the inner workings of them can be revealed, allowing attackers to get access to private data. Meltdown exploits a race condition, which happens when the CPU is handling instructions, between memory access and privilege checking. This CPU design flaw gives a process the ability to bypass normal privilege checks. This means an attacker can read data from anywhere in a computer's memory, even the CPU's cache[1].

Another example of a timing attack is Nemesis, which went unnoticed for years and takes advantage of a feature of microarchitecture [2]. This side-channel attack exploits the timing behavior of the CPU's interrupt mechanism, compromising the security of the hardware. The Nemesis attack functions by measuring the amount of time it takes for a timed interrupt to occur, allowing attackers to deduce that instructions are executed in the hardware-enforced enclaves.

B. Mitigation Techniques

A mitigation technique for timing attacks is constant-time algorithms [1]. If all algorithms are running at the same time, attackers are not able to uncover clues on the timing of different operations, keeping the system secure. However, this mitigation comes with a drawback. If every execution needs to run in constant time, then every operation can only run as fast as the worst-performing operation. Ultimately, this lowers the efficiency of the algorithm. Another mitigation technique against timing attacks is masking. In this technique, a secret is split into many shares, so the attacker needs to gather and put together all of them to reveal the shared secret [1]. The downside to masking is that it's only practical in certain algorithms with a suitable algebraic structure.

IV. Acoustic Attack

A third category of side channel attack is acoustic attacks. The acoustic side-channel attack is when the attacker measures the sounds produced by a device. These attacks have been performed by reconstructing a user's keystrokes from an audio recording of the user typing. From analyzing these unique sounds from the keys, anyone with the right resources can decode the precise letters and numbers being typed. Attackers can also get this information by listening to the sounds emitted by electronic components.

A. Execution of Acoustic Attacks

In an experimental study, a team successfully developed a deep learning-based acoustic attack. This attack was employed to classify laptop keystrokes, utilizing audio recordings that were captured from a phone's microphone [3]. After extensive training on keystroke patterns, this strategy achieved a 95% accuracy rate. This model was further trained using recorded keystrokes from the video conferencing software, Zoom, which yielded a 93% accuracy rate. These results prove the practicality of these side-channel attacks with proper equipment and algorithms.

B. Mitigation for Acoustic Attacks

Some of the mitigations to the acoustic side channel attacks are changing the typing style. It is also recommended to use randomized passwords and passwords that do not contain full words to make it more difficult to decipher. Using randomized and stronger passwords means that the sequence of characters in the password will not be easily predictable or based on common patterns. These attacks rely on the recognizable sound patterns from typing common words. When passwords are randomized, the attacker cannot rely on recognizing specific typing patterns associated with common passwords which makes it harder to decipher the password. You can also add randomly generated keystrokes for voice call based attacks.

Two other mitigations include sound-free keyboards and keyboards with keys that will produce the same sound. The issue with sound-free keyboards is that they can be expensive and would take time to get used to. Additionally, it is not known if creating keyboards that produce the same keys are possible. It is also not known if it is possible to construct such keyboards and how they will perform over time given wear and tear.

V. Power Attack

Power analysis is a common avenue into attacking a system, this attack is done by monitoring the power consumption of a cyber-physical system while executing an operation. During

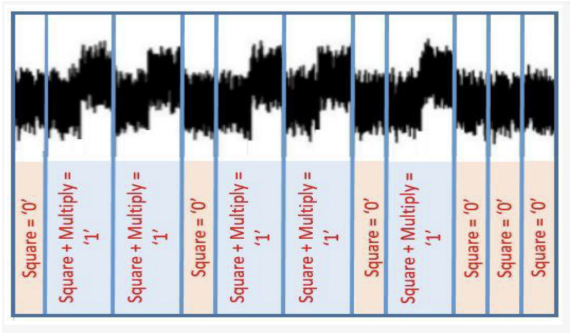

Fig. 2. An example of an oscilloscope reading that provides visual representation of the difference in power consumption between a multiply operation and square and multiply operation. [7]

any operation done by a system the amount of electric charges change, and by measuring these adjustments an attacker is able to obtain information that can lead to uncovering sensitive data and operations. All of which pose serious security threats to the system. The two most common analysis approaches are Simple Power Analysis (SPA) and Differential Power Analysis (DPA). SPA is more generally known as the visual attack in the sense that the information gathered is physical and are visual readings of power. This form of analysis is more inclined to reveal cryptographic operations rather than extracting secret keys from an operation [6]. Whereas DPA uses statistical analysis to average subsets of traces with a given assumption made about the operation and then finding the differences between them. If the difference are high enough then the assumption made is likely to be valid [7]. Whether the assumption was about secret keys or the execution of a specific operation, DPA has now given the attacker confidential information.

A. Execution of Power Attacks

Attackers will take information like a surge in power or inversely a drawback in power and uses this information to run either SPA or DPA. This data is generally gathered by use of an oscillator. This form of data gathering is especially useful to SPA as it visually displays power consumption. The patterns in this data may be used to figure out sequences of operations. A notable implementation of this attack has been its ability to figure out a RSA key. During this attack if the oscilloscope reads the binary bit as '1' then the operation naturally relays a square and multiply function. If the bit reads '0' only a square operation is performed [7]. We know that a simple square function takes less energy or power consumption than a square and multiply function, given this information an attacker now knows the binary sequence of the private key exponent of the RSA encryption.

B. Mitigation Techniques

While low power usage and noisy systems help deter attackers from gaining information off a system, these precautions have workarounds. Since this form of attack is generally passive and non-invasive, normal precautions like air gapping and auditing a system will not give sufficient security. While both of the previously mentioned mitigations are important to a system in general, in this case they give zero to no protection to Power Attacks. This is due to the non-invasive nature of this attack. Auditing a system will have no way of detecting any foul play as nothing in the code or cyber-physical system is being altered. A general rule of thumb when trying to prevent this type of attack is ensuring that the system does not reveal when a new or different operation is performed. Another technique is computing pieces of a key in an arbitrary order in order to mix up the power tracing. In theory this makes finding the order of decrypting a secret key more unpredictable. Instead of reading voltages and following the shown operation order, attackers now have to also figure out the correct sequence in which the system should have performed these operations in order to get cohesive information. As mentioned previously, noise is not a foolproof mitigation but it does force attackers to spend more time gathering information because of extra information needed to be weeded out. Typically an extra step of averaging information gathered must be done in order to accurately use any measurements gathered. This mitigation however this mitigation does not work on SPA's and not DPA's due to DPA's ability to drown out arbitrary noise.

VI. Setup

After receiving the hardware package from CSAW, we assembled the Arduino Uno and added the provided attachments to the board. We made use of a dongle to allow the use of USB to USB-C for our laptops, which were mostly of Mac OS origin. We were able to set up the Arduino software using the instructions provided on the CSAW Github in the ReadMe file in the Hardware folder. The only challenge we faced here was initially we didn't understand that we had to download Avrdude, but once we were able to successfully install it, everything ran smoothly.

VII. All White Party

A. Initial Assessment

The All White Party challenge was based on being invited to a party. The goal was to get past a security system that is asking for a username and 10-digit password PIN credentials. In the challenge description, there was an emphasis on time which gave the impression that this would be a timing side channel attack.

B. Approach and Methodology

Initially, the first idea was to put in a random word just to see what the output would be when trying for a username. After putting in the random username there was a vibration that came from the Arduino. There was a vibration that came after any attempt for the username. After figuring this out, the next guess was that maybe there would be a time difference in the vibrations depending on whether it is an incorrect or correct input. For example, if there is a lag in the vibration, maybe it is a correct letter. The next step in this process was

to go through each letter to see if there was a time difference in the buzz after the attempts.

After going through each letter and waiting for the vibration the team was unable to detect the time difference just by feeling it. Because of this, the next step was to use the Toggle timestamp function in the Arduino IDE to see the time difference. Unfortunately, this also did not help because it depends on when the letters are inputted and it is not true to the actual time the vibration comes after putting in a letter.

At first, the team did not take into account if the username was case-sensitive and only checked for lowercase letters. In order to solve the earlier issues, there was an attempt to try to find a way to check for the exact timing of each vibration since the Toggle Timestamp was not reliable.

C. Results

After trying uppercase letters and trying to find some sort of way to get the exact time between the attempt and the vibration, unfortunately, we were still unable to figure out the correct username to get into the All White Party. **This challenge was never completed.**

VIII. Bluebox

A. Initial Assessment

When this challenge is run on the Arduino, 4 tones are played right after each other. Sometimes only two or three tones are outputted, signifying that the same digit is pressed twice or three times in a row. The serial monitor prompted the user to input a 4-digit pin, entering each digit one at a time. The initial assessment for this challenge was that this is an acoustic side-channel attack and that the tones that are outputted correspond to the digits on the keypad. Additionally, after each failed attempt at the 4-digit pin it would change in each iteration making it more difficult to get correct, as only one attempt was given to get the code correct.

B. Approach and Methodology

The approach taken to pinpoint the 4 digit pin was to map each digit to a tone and decipher which tones are being played in the 4-digit pin. In order to map the tones to the digits, each input on the keypad was played one at a time. A pitch checker app was used to record what note was being played for each digit and the frequency it was associated with. After figuring out the mapping for each digit to its associated tone the audio played from the Arduino was recorded on a phone. This recording was then played back to the pitch checker, finding each note that is played. These notes are then deciphered by referencing the mapping sheet and finding which keypad digit is associated with that tone.

C. Results

After a couple of attempts, the four-digit pin was accepted, asking for an 8-digit pin to reveal the flag. However, there were some issues with the 8-digit pin. Using the same method for revealing the 4-digit pin, the result would be 'B339B009'. We also spliced a video together of the hidden pin followed by the

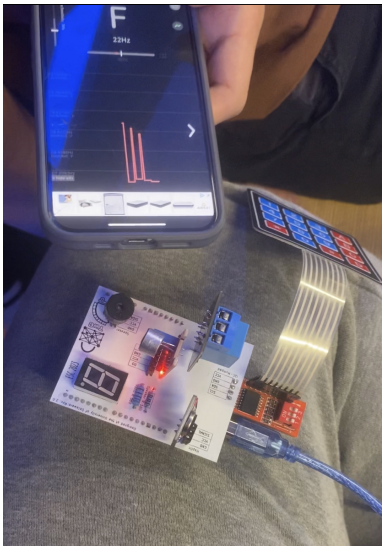

Fig. 3. Setup of Arduino with pitch checker app held up to speaker to identify the tones played

1	hihiA#	953
2	B	1011
3	C	1067
4	D	1189
5	D#	1249
6	E	1308
7	F	1427
8	F#	1487
9	G	1542
0	A	1722
A	C#	1130
B	F	1365
C	hihiG	1601
D	hihihiA#	1837

Fig. 4. The mappings of each digit on the keypad: the first column is the inputs on the keypad, the second column is the note outputted, and the 3rd column is the frequency of the tone

tone we thought to be correct and checked their tones using a frequency website, further solidifying our proposed pin. However, this code did not work. This led to the conclusion that some digits were mapped incorrectly, but there was no clear conclusion as to why the mappings were incorrect. Upon further review, the codes of 4-digit pins that passed are highlighted in yellow in Figure 2. These are the digits that are assumed to be mapped correctly because the 4-digit pin was accepted, asking for the 8-digit pin. The numbers assumed to be correct were 1, 2, 4, 5, 6, 9, 0.

After attending the CSAW competition, it was uncovered that the 8-digit previously found ('B339B009') was correct the whole time. The combination not being accepted could be associated with buggy Arduino hardware. **This challenge was completed before the competition.**

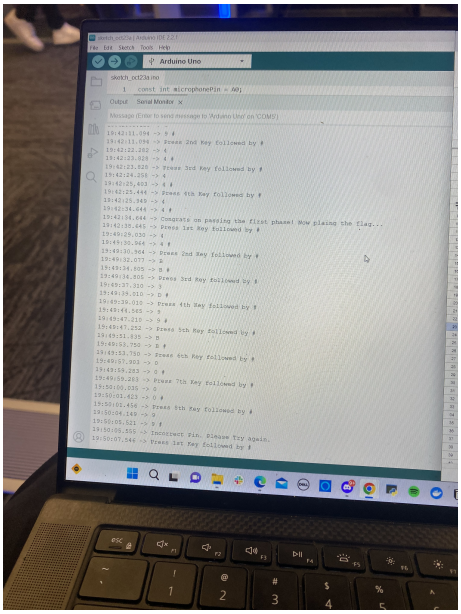

Fig. 5. One of the correct 4-digit pins accepted, but the 8-digit pin was not accepted

IX. Operation SPItFire

A. Initial Assessment

Glancing at the problem description of SPItFire we see that the first letters are capitalized. Unfamiliar with what SPI is a quick search into this topic was done in order to determine if this is a clue into how to solve this challenge. It was found that SPI stands for Serial Peripheral Interface and essentially is a communication protocol used between a variety of embedded systems. This form of communication has a master-slave relationship where the microcontroller sends instructions to the display of the Arduino and requires a clock [1]. Using this information one can infer that this is the form of communication that is being used by Arduino.

B. Approach and Methodology

Based on our research we were reminded that information is sent in the form of binary. This makes more sense in terms of the audio being outputted by the Arduino Uno; initially, we thought the sounds coming from the Arduino represented the dots and dashes that create Morse code, so we attempted to first play the audio into a decoder, but we received a sequence of E's and T's, as shown in **??**, in return. After the challenge denied this first effort, we tried to write out the sequence ourselves, but after deciphering it to, "IISEESTRM" and receiving the same 'incorrect header' error message, as well as hearing the same sequence once again, we decided it was most likely not Morse Code.

We then reverted back to the fact mentioned earlier that information between systems is sent in binary and took that approach. Additionally, in the description, it says we are spies and are assigned a mission of deciphering so given this piece of information we decided to work even further into the possibilities with binary. First, we decided to allow the shorter sound to be a 0, and the longer to be a 1. We also set all breaks in the audio to be 0s. Doing this and repeating it for some time, we came up with the following binary sequence:

1010 0101 0001 0101 0010 0010 0010 1010 0110 0100 1100 1001 1100 1110

Using this result and computing the binary into a Hexadecimal value we were able to obtain the header that the challenge refers to in the serial monitor. Although the challenge doesn't directly confirm that the header is correct when we weren't entering 'A5', we received an error stating 'incorrect header', but after beginning our response with A5, or even entering A5 alone, we received a new error: 'Incorrect length". This confirmed to us that the header was A5. We also checked similar combinations like B6 and C2 to ensure it was specific to A5 and we were indeed correct. After this A5, we began to do research and came across the A5/1 ciphering algorithm[5], which we soon found out is used to provide privacy in over-air communications. After finding this out, we were excited and believed this could possibly have been a part of the challenge. The A5/1 algorithm requires a 64-bit binary key to encode a message. Unfortunately, whenever we tried different sequences of 64 bits, we still weren't able to gain the flag by following this path.

After thinking more about what the error message "Incorrect Length" could mean, we came up with that it follows the header. This is realized based on how the message "Hello" was received. In hex, "FLAG" is "46 4C 41 47", which is 4 bytes long. Therefore the length would be '04' added after the header 'A5'. This would give 'A504464C41470A' as the input for the serial monitor. However, when this was inputted another error message popped up, "Bad CRC".

```
19:09:19.119 -> Recieving message "HELLO"...
19:09:33.129 -> Please send the message "FLAG" in hex (over serial
19:11:33.873 -> Error: Bad CRC
```

Fig. 6. Fig. 9: Bad CRC error mesasge given

CRC stands for cyclic redundancy check, an error-detecting code that is used to determine if a block of data has been corrupted. A mathematical operation in required to find the CRC; however, there are websites online that can perform this operation. The hex value that has been found to be correct so far is 'A504464C4147'. When this is inputted into the calculator, it gives a CRC of 'DA'.

The string, 'A504464C4147DA' was inputted into the serial monitor, and this completed the challenge.

C. Results

Overall we made use of Morse Code, Binary, Hexadecimal, SPI and began to look at the A5/1 algorithm. The solution to this challenge was a string including the header, message, and crc. Using the error codes and putting all this information together helped figure out the string, 'A504464C4147DA', which was the flag. **This challenge was completed after the competition.**

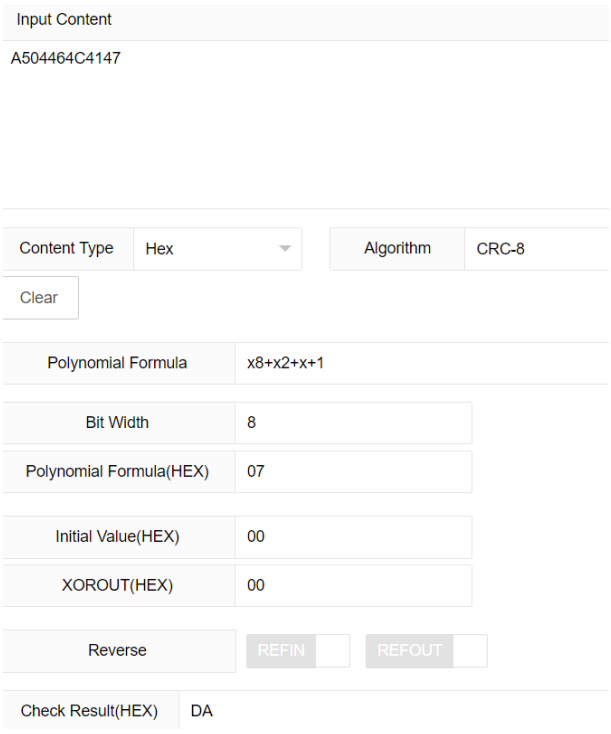

Fig. 7. CRC value found from online calculator

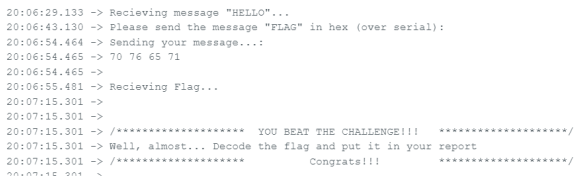

Fig. 8. Serial Montior screen showing challenge was beat

X. czNxdTNuYzM

A. Initial Assessment

At an initial glance of this specific problem, one thing noticed was that the name seems to be a string of letters in upper and lower cases. All of the previous challenges have had some sort of hint given in the name so we assume that this one also does but we are unable to make sense of the order or significance of the name. Additionally, when running this challenge the Serial Monitor outputs the following string, ”QSBzb3ByYW5vIG9mIHNvdW5kLCByZWFjaGluZyBmb3IgdGhlIGhlYXZlbnMu”. Given in the description is ”cryptic dance”, ”harmonize”, ”symphony”, ”rapid”, and ”swift”, these keywords signify that the name of the challenge and then the string given is encrypted and that there possibly is another timing attack. As for other side channels, there is no indication that it is an acoustic attack even though this challenge is music-related. However, this still leaves one feeling uneasy as there are no other leads on how to approach this problem.”

B. Approach and Methodology

After more deliberation and thinking about our SPItFire problem where the thought was that the encryption used there was a5 we decided to look into other forms of encryption and started inputting the name of this challenge into a couple of them to see if any resulted in something legible. We came across base64 encoding and learned the encoding process and decided to try it [2]. To our surprise after putting czNxdTNuYzM into a base64 decoder we got back the string ”s3qu3nc3” It is important to note that we used an encoder easily accessible on the internet rather than trying to make a Python script for it. After finally feeling as though we were getting somewhere we put in what the serial monitor had outputted and got back the string, ”A soprano of sound, reaching for the heavens.” This made a lot of sense to us since some of the buzzwords from the description were music related. Although we still had no idea what the following sequence was we then knew that the sequence is most likely a music-related phrase that has been decoded by base64. After a quick Google search, we realized that reaching for the heavens is a song and decided to take the name of the composer, Gerald Cohen, and encrypt it using base64 getting, ”R2VyYWxkIENvZWhu”. This however was not correct and neither was trying other encoded words like bass or tenor.

The next approach taken was directly related to the Arduino module that outputted numbers on its screen. The screen had 3 different possible outputs. It was either a number (1-9). an underscore (_), or a period (.). The point of the challenge is to find the next number in the sequence. Therefore, the number being outputted on the display are the numbers in the sequence leading up to what needs to be inputted into the serial monitor. The underscore in this case represents the separation between each digit in a number. For example 120 would be outputted like 1_2_0. The periods are basically serving as a comma that separates each number in the sequence from each other. In order to find the sequence, the screen of the arduino module was recorded in slow mo using a phone and the results were recorded. The sequence found was 1, 2, 6, 24, 120, 20, 140, 1120, 10080, 1008, 11088, 924, 12012, 858, 12870, 205920, 3500640, 194480, 3695120, 184756, 3879876, 176358, 4056234. In order to find the next digit in the sequence, a sequence calculator was used.

C. Results

From the approach taken the results came out to be ’97349616’, as it was the next number that came in the section. When this number was inputted into the serial monitor the flag was revealed. This seems to be a visual timing attack as the code was able to be uncovered from observing the numbers on the Arduino module screen to discover the next number in the sequence. **This challenge was completed after the competition.**

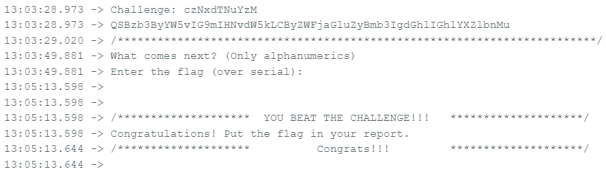

Fig. 9. Flag revealed for czNxdTNuYzM challenge

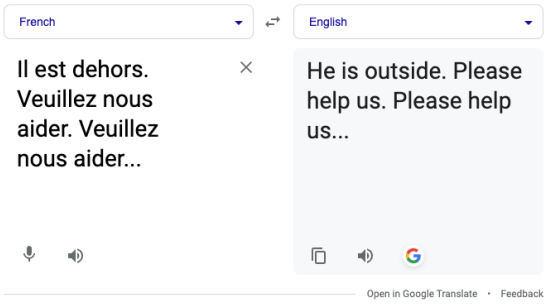

Fig. 10. Serial Monitor Output translation using Google Translate

XI. Sock and Roll

A. Initial Assessment

The description of Sock and Roll gave very little to work off of. Historically there has been a lot of information to pull from or clues to have somewhere to start from the challenge description but in this case there were not many. After running the hex file the Arduino Uno plays an almost screech like tone that incidentally also sounds like a bomb getting ready to self destruct. One also sees that the Serial Monitor out puts a message that seems to be in French. After translating the phrase we find that the message reads, ”He is outside, Please help us. Please help us...”, quite frightening but at least there is confirmation that the Happy Tap Dancing Socks Message Machine 2000 is working.

B. Approach and Methodology

After taking a closer look at all of the components we decided that this may be a timing attack because of the urgency portrayed in the challenge description. However, it was later discovered that this was in fact an acoustic attack. There was the high pitched noise as well as moments of silence, which was thought to be morse code. The sound made from the Arduino was recorded. The noises were treated as dots and the silence was treated as dashes. After writing done the code and inputting it into a morse decoder the resulting output was ”Socks”. Once remembering that the purpose of the challenge was to communicate a distress signal to the outside world we noticed that by taking out ’ck’ from ’socks’ leaves ’sos’.

C. Results

In order to retrieve the results to this challenge, the speaker piece on the Arduino had to be removed when ’ck’ was played, resulting in the message of ’SOS’. This was done by simply unplugging this piece. Once doing this successfully,

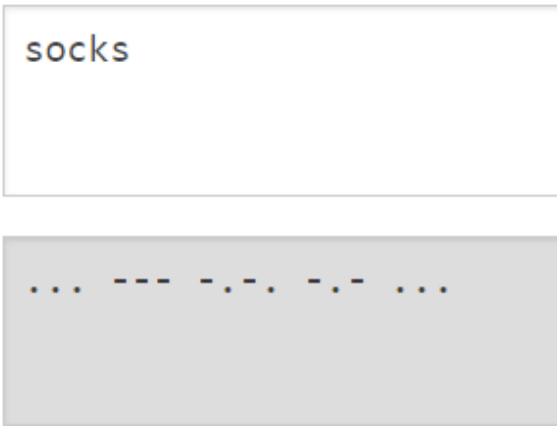

Fig. 11. The morse code translation of the word Socks

this revealed the flag. This was an acoustic attack, as the Arduino outputted noises that was able to be used to ultimately uncover the flag. The morse code noise that came from the microphone was used to decode the message being sent. **This challenge was completed after the competition.**

XII. Vender Bender

A. Initial Assessment

For the Vender Bender challenge, the goal is to get some free snacks from the vending machine. The mission involves manipulating the vending machine by jamming it to get the free snacks. Initially, there was a message received indicating the successful execution of the motor movement, which would repeat periodically. Subsequently, a buzz sound would come up, followed by a message that prompted to put in ”ERR” in the serial monitor to jam the machines. This would continue in the serial monitor at random moments. The process started by putting the message ”ERR” in the serial monitor to see what would happen, but nothing new came up and the machine continued as normal. The initial idea was that the message ”ERR” should be put in at a certain time.

B. Approach and Methodology

Upon hearing the buzz signal, the team promptly entered the ”ERR” command into the serial monitor. This action resulted in a successful outcome, accompanied by a message stating ”Motor Error 5902 Reported.” The challenge required the team to replicate this process four more times, ensuring a total of five consecutive and successful attempts.

This phase of the challenge demanded persistent effort from the team. Numerous trials were conducted, continuously trying to attain the desired outcome. It took the group several attempts, but after successfully completing the five required attempts, the desired message was received indicating the successful completion of the challenge.

C. Results

After successfully completing the challenge, the serial monitor showed the message ”You Beat the Challenge!” with the

flag: mMmCaNdY shown in Figure 7. **This challenge was completed before the competition.**

```
00:34:13.659 -> Motor movement SUCCESS. Snack was dispensed for $2. Insert another credit for a new snack.
00:34:18.670 -> After credit is reciever, send "ERR" to jam the motors.
00:34:22.182 -> Motor movement SUCCESS. Snack was dispensed for $2. Insert another credit for a new snack.
00:34:27.178 -> After credit is reciever, send "ERR" to jam the motors.
00:34:30.653 -> Motor Error 5902 Reported. Slight but not signifigant motor movement detected. Retry Attempt 1/5
00:34:33.076 -> Motor Error 5902 Reported. Slight but not signifigant motor movement detected. Retry Attempt 2/5
00:34:35.013 -> Motor Error 5902 Reported. Slight but not signifigant motor movement detected. Retry Attempt 3/5
00:34:36.758 -> Motor Error 5902 Reported. Slight but not signifigant motor movement detected. Retry Attempt 4/5
00:34:38.640 -> Motor Error 5902 Reported. Slight but not signifigant motor movement detected. Retry Attempt 5/5
00:34:39.660 ->
00:34:39.660 ->
00:34:39.660 -> /********************  YOU BEAT THE CHALLENGE!!!   ********************/
00:34:39.660 -> Place the following flag in your report
00:34:39.660 -> mMmCaNdY
00:34:39.660 -> /********************          Congrats!!!         ********************/
00:34:39.660 ->
00:34:39.660 ->
00:34:55.704 -> After credit is reciever, send "ERR" to jam the motors.
00:34:58.176 -> Motor movement SUCCESS. Snack was dispensed for $2. Insert another credit for a new snack.
00:35:03.212 -> After credit is reciever, send "ERR" to jam the motors.
```

Fig. 12. Flag revealed after successfully beating the challenge

ACKNOWLEDGMENT

We'd like to extend a special thank you to our team advisor Andrew Zeliff of Georgia Tech Research Institute for answering our questions and talking us through multiple different scenarios.

REFERENCES

[1] S. Campbell, "Basics of the SPI communication protocol," Circuit Basics, https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/ (accessed Nov. 2, 2023).

[2] "Base64," Wikipedia, https://en.wikipedia.org/wiki/Base64 (accessed Nov. 6, 2023).

[3] "Morse Code Decoder," Morse Code.World, https://morsecode.world/international/decoder/audio-decoder-adaptive.html (accessed Nov. 1, 2023)

[4] "Base64," Wikipedia, https://en.wikipedia.org/wiki/Base64 (accessed Nov. 6, 2023).

[5] "A5/1 algorithm implementation", https://ericszla.wordpress.com/2017/05/13/a51-ciphering-algorithm-implementation-in-c/ (accessed Nov. 6, 2023).

[6] J. Lake, "What is a Side Channel Attack? (with Examples)," Comparitech, Apr. 16, 2021.

[7] Jo Van Bulck, F. Piessens, and Raoul Strackx, "Nemesis," Oct. 2018, doi: https://doi.org/10.1145/3243734.3243822.

[8] J. Harrison, E. Toreini, and M. Mehrnezhad, "A Practical Deep Learning-Based Acoustic Side Channel Attack on Keyboards", 2023.

[9] Cloudflare. "Meltdown and Spectre - Computer Vulnerabilities." https://www.cloudflare.com/learning/security/threats/meltdown-spectre/

[10] Gregor Haas and Aydin Aysu, "Apple vs. EMA: Electromagnetic Side Channel Attacks on Apple CoreCrypto", 2022 https://eprint.iacr.org/2022/230

[11] L. Sun, "Side-channel attacks: How differential power analysis (DPA) and Simple Power Analysis (SPA) works," AnySilicon, https://anysilicon.com/side-channel-attacks-differential-power-analysis-dpa-simple-power-analysis-spa-works/ (accessed Sep. 15, 2023).

[12] J. Treus and P. Herber, "Early Analysis of Security Threats by Modeling and Simulating Power Attacks in SystemC," 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), Antwerp, Belgium, 2020, pp. 1-5, doi: 10.1109/VTC2020-Spring48590.2020.9129426.

[13] YongBin Zhou and DengGuo Feng, "Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing", State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, https://csrc.nist.gov/csrc/media/events/physical-security-testing-workshop/documents/papers/physecpaper19.pdf

[14] Das D, Sen S. Electromagnetic and Power Side-Channel Analysis: Advanced Attacks and Low-Overhead Generic Countermeasures through White-Box Approach. Cryptography. 2020; 4(4):30. https://doi.org/10.3390/cryptography4040030