Building an Integrated Station for Implementation Security Validation

Final Report

Project Members:
Alan Lin (#101036660)
Ben Bozec (#101034327)
Brannon Chan (#101045946)

Group No. 43

Project Supervisor:
Professor Mostafa Taha, Ph.D.

April 14th, 2021

Department of Systems and Computer Engineering
SYSC4907A - Fall 2020 to Winter 2021

Table of Contents

Table of Contents 1

Abstract 3

Chapter 1
Introduction 4

Chapter 2
Professional Practice 5

Health and Safety 5
Engineering Professionalism 5
Project Management 5
Project Roadmap 6
Justification of Suitability for Degree Program 7
Individual Contributions 7

Chapter 3
Initial Proposal 9

Original Design & Background 9
Proposed Development Roadmap 10
Current Development State 10

Chapter 4
Project Redesign 11

Refined Operation 12

Chapter 5
Technical Specifications 14

Hardware & Software Components 14
AES128 & Power Analysis Techniques 16
Advanced Encryption Standard (AES) 16
Differential Power Analysis (DPA) Attack 17
Correlation Power Analysis (CPA) Attack 20
Test Vector Leakage Assessment (TVLA) 22
Software Sequence Diagram 24
High-Level System Circuit/Component Diagram 25
Internal Project Data & Testing Management 26

Chapter 6
Conclusion 28

Chapter 7
Reflections **29**
Group Reflections 29

References **31**

Appendix **33**
Project GitHub Repository 33
AES/DES TVLA Dataset 33
Building an Integrated Station for Implementation Security Validation 34
Project Proposal 34

Abstract

In this Internet of Things age, we have witnessed an unprecedented surge in development and commercial deployment of IoT edge devices, low-power single board computers (SBS) and similar embedded devices; many of which hit the market without sufficient security measures. However, due to the distributed and physical nature of such devices, physical security is not provided/guaranteed in most use cases. As such, it becomes viable for adversaries to perform Side Channel Analysis (SCA) to compromise the target device [1].

To begin exploring this field of physical security, we focus on examining Advanced Encryption Standard (AES) with Power Analysis to better understand their vulnerabilities and software countermeasures. This project seeks to create an integrated implementation security validation station to analyze and validate the security of common AES-128 implementations via Power Analysis techniques such as DPA, CPA and TVLA. The validation station relies on the use of a NewAE ChipWhisperer for data collection and our SimpleGUI written in Python & C to configure, calculate and display results.

Chapter 1
Introduction

In layman terms, Power Analysis is the act of scrutinizing the input and output power of a device while it completes a cryptographic operation and extracting the secret data contained within the device using the recorded power measurements [2]. To tackle these software vulnerabilities and algorithm implementation flaws, we propose a low-cost, user-friendly, plug and play pre-assembled power analysis tool; the Integrated Security Validation Station. In conjunction with SimpleGUI (the ChipWhisperer's Linux-based UI for user control), one can now perform previously complex and highly technical power analysis attacks such as Differential Power Analysis (DPA) and other techniques against an implementation of AES-128.

Chapter 2
Professional Practice

Health and Safety

As the vast majority of our system is limited to software, the concern of health and safety was a minor consideration in the development of our project. The sole piece of hardware that we utilize does not present any considerable risks or dangers to operate, so we feel that no further steps are needed to ensure safe usage by users. To date, we have not found reason to believe that any action is required to ensure that our system can be safely operated by both ourselves and by product users.

Engineering Professionalism

In order for our team to accomplish goals and complete quality work, engineering professionalism is a key focus. In any engineering project, ensuring our work is safe and ethical for the public is a must. Practicing engineers in Ontario need to regard their work to the public welfare as paramount [19, A-3]. Though we are not engineers yet, we will still follow the same ethics. To do this, our team worked with tight communications channels, used professional documentation and sought only reputable research materials. To review our work for correctness, we requested reviews and clarification from our qualified supervisor Mostafa Taha. Considering our project intends on testing and displaying potential security vulnerabilities in encrypted systems, we will not use our findings in any malicious ways. The target devices we used were designed for testing and education purposes, but can be applied to practical devices such as a smart card reader. We will take our knowledge of this topic to better understand and teach others about potential vulnerabilities of power analysis in everyday devices. We will continue to practice engineering professionalism as we follow the path of becoming a professional engineer.

Project Management

For our team to reliably manage the work of this project, both scheduled and ad hoc meetings/discussions were done and professional project management tools were used. Biweekly meetings were scheduled with our supervisor to discuss progress and inquire about any specific topics that needed further clarification. The frequency of meetings would ramp up to weekly when milestone deadlines approached and the output of work was increased. Among ourselves, when working on the project, we would also hold prompt meetings to provide immediate technical support. In terms of project management tools, for the software related work, GIT was used for the version control. Our team consists of three members and GIT allows us to simultaneously produce work without high risk of work merging conflicts. For ease of collaboration in the documentation aspects, Google Docs and Microsoft Teams were heavily used. Microsoft Teams also involved discussions with the supervisor.

The approach we took to tackle this project was done in steps to break the project into smaller pieces. Such that we worked our way up to the final product while maintaining consistent delivery of work. Distribution of the planned work was done in the project proposal stage and any other allocations of work were managed in our meetings. The importance of following project management practices such as meetings and tools is evident from our experiences and we strived to ensure our project was professionally managed.

Project Roadmap

The milestones are depicted in the Gantt chart below in *Figure 1*. It covers the development plan for the completion of the system as well as time allotment to prepare for the oral presentation and to write the Final Project Report. These milestones are further elaborated on and divided into sub-goals beneath *Figure 1*. This timeline was updated in our progress report as we believed it was more feasible than the original, as it accounts for delays that we met during earlier development and allows for more time to develop the core functions of the system.

Figure 1: Gantt chart depicting the development plan for the remaining milestones.

Power Analysis Software Development (January 18th – February 19th, 2021)

- Finalize implementation of TVLA evaluation system (January 18th - January 25th)
- Expand selection of power analysis methods and default algorithms for testing (January 25th - February 12th)
    - TVLA Analysis complete by January 25th
    - DPA Attack complete by February 1st
    - CPA Attack complete by February 8th
    - Add options to GUI for each power analysis method (e.g., Initial trace amounts for TVLA) [February 9th - February 11th, 2021]
- Improve the results component of the GUI (February 15th - February 19th, 2021)
    - Display results that are relevant to the chosen power analysis method
- Final Testing of Software (February 8th - February 19th, 2021)
    - Manual testing (Performed throughout)
    - Writing Test cases

Oral Presentation (March 1st – March 19th)

- Oral Presentation preparation/practice
- Give Oral Presentation (March 15th – March 19th)

Final Project Report (February 12thth – April 14th)

- Write Final Report Draft (February 12th - February 26th)
- Review, Edit and Finalize Report (March 20th - April 14th)

Justification of Suitability for Degree Program

The three members of this project are students in the Computer Systems Engineering stream. This degree explores the combination of hardware and software, focusing on the implementation of embedded and integrated systems. Among the topics we studied on embedded and integrated computer systems was security. We learned the means by which individuals with malicious intent could attempt to gain access to these systems, as well as the countermeasures in place to protect against such attempts. Our project involves one such security risk. In the future, when we work on integrated systems, side channel power analysis attacks will be a problem that we must consider. Through our work, we have also made use of our knowledge and skills in programming hardware through the integration of the ChipWhisperer board into our system. This includes general use of programming languages such as python and C, as well as knowledge of computer architecture needed to understand the process by which these attacks are performed.

Individual Contributions

Member	**Project Contributions**	**Report Contributions**
Alan Lin	● Learning and group support of ChipWhisperer Lite usage ● Implementation of CPA ● Supported DPA implementation ● Supported TVLA implementation ● Core GUI development and testing ○ Integration of CPA, DPA and TVLA to GUI ● Testing of CPA, DPA and TVLA attacks/analysis ● Investigated ‘custom’ algorithm	Engineering Professionalism, Project Management, Hardware & Software Components, Differential Power Analysis, Software Sequence Diagram, Conclusion, Reflections

	implementation	
Ben Bozec	● Investigated custom algorithm implementation ○ Tested multiple means to upload algorithms to test board ○ Researched integration of SimpleSerial communication protocol ● Added support for ARM target processor ● Testing of CPA, DPA, and TVLA attacks/analysis	Introduction (P.2), Health and Safety, Justification of Suitability for Degree Program, AES128 & Power Analysis Techniques Intro, Correlation Power Analysis
Brannon Chan	● Completed preliminary research for project background knowledge and state-of-the-art hardware security research ● Ported over the current TVLA implementation for use on the ChipWhisperer Lite ○ Acquired basic knowledge of ChipWhisperer Library functions/code as a result ● Assisted in hardware & software design of Integration Validation Station in cooperation with team members ● Attempted to assist in custom cryptographic algorithm implementation, though the feature was ultimately scrapped	Abstract, Introduction, Initial Proposal, Original Design & Background, Current Development State, Test Vector Leakage Assessment (TVLA), Project Redesign.

Chapter 3
Initial Proposal

Our originally proposed integrated security implementation validation testing station promised more features and encompassed a much larger scope of power analysis use cases/applications. Built around a NewAE ChipWhisperer Lite (32-bit ARM or XMEGA), this open-source board would allow for a variety of SCA and power-based attacks to be performed without the need for additional analog hardware (oscilloscopes or other signal processing devices). Since a majority of development in this project would lie in it's software component, we aim to produce a novice & expert friendly GUI for performing power analysis techniques using the existing ChipWhisperer Python libraries to provide largely automated measurement and power trace processing capability.

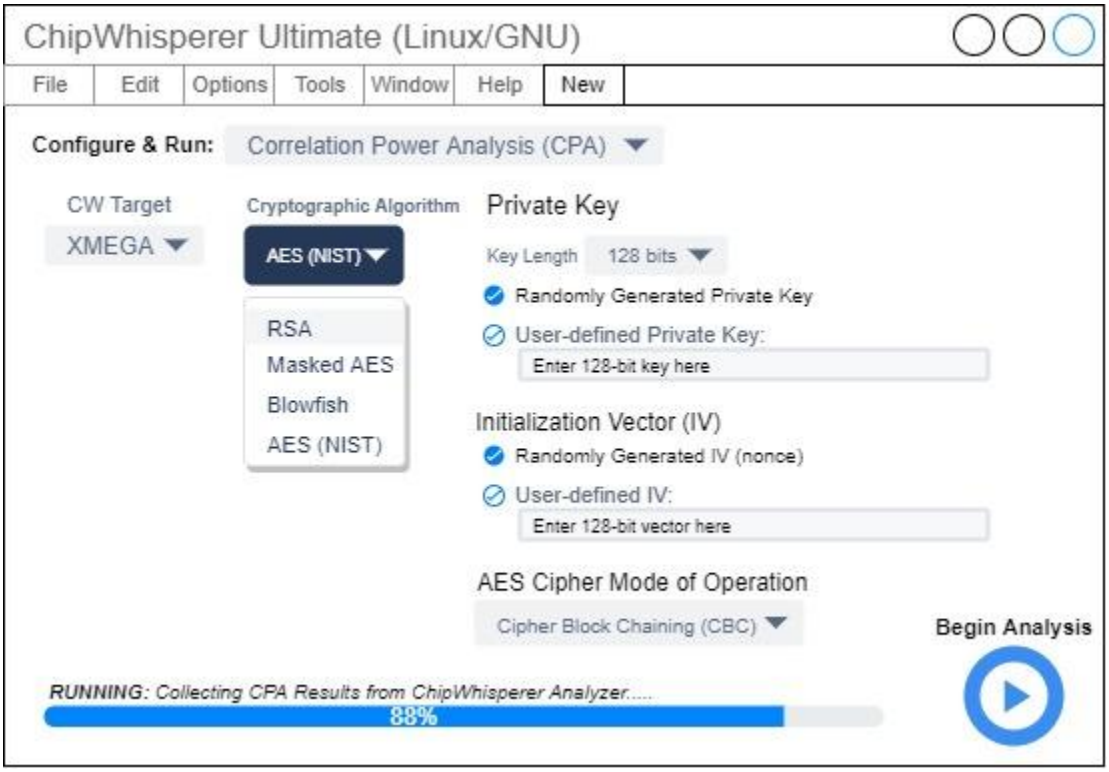


Figure 2: Proof of concept of project's GUI with initial planned features

Original Design & Background

The original validation station design was supposed to be compatible with ChipWhisperer evaluation boards, FPGAs, or other externally connected microprocessors & microcontrollers, and would be able to assign an overall "security" score to the Device under Test (DUT) and Software under Test (SUT). Additionally, the station should theoretically be able to perform DPA, CPA, and TVLA on symmetric/asymmetric encryption algorithms, and any other existing cryptographic algorithms by extension. It is worth noting that the system functionality and objectives listed in the original proposal (see Appendix – Submitted Project Proposal:

Objectives & Project Functionality) will have to be revised and scrutinized, due to the vagueness in the original design, its implementation, and real power analysis operation.

Proposed Development Roadmap

Below are the original development phases and Gantt Chart (*Figure 3*) adapted from the submitted Project Proposal (see *Appendix – Submitted Project Proposal: Timetable*), based on the originally defined objectives and functionality.

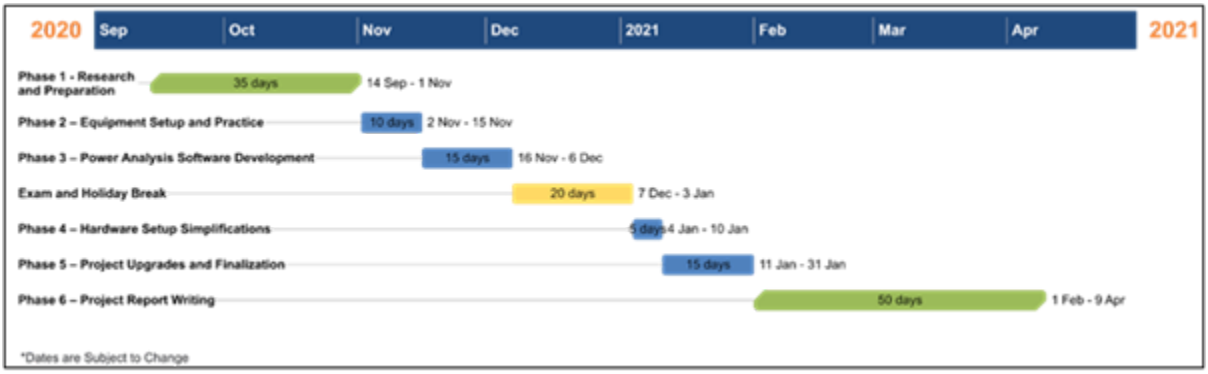


Figure 3: Graphical Timeline of Major Milestones

Phase 1: Research of Side Channel Analysis, Power analysis techniques and Original Project Proposal report submission.
Phase 2: Acquisition, Setup and Familiarization of ChipWhisperer boards and ChipWhisperer Python Libraries.
Phase 3: Creation of Simple Power Analysis software for CW Boards, Calculation of collected data and overall "security" score, easily readable graphs/data from results, Testing of system's hardware and software components.
Phase 4: Simplification of Design and Physical Implementation of the Security Validation Station, Cleanup of physical connections and overall package.
Phase 5: Improved GUI for user experience, adding integration for 3rd party software support, Submission of Progress Report, and Finalize H/W and S/W setups via Testing.
Phase 6: Write, Review, and Rewrite both Draft and Final copy of Final report.

Current Development State

In working on the project, we have encountered multiple hurdles that have impacted our ability to deliver on implementation/completion of project objectives and features previously outlined in the initial Proposal. Because of the lack of sufficient support and proper documentation for the ChipWhisperer Python Library, objectives and features relying on this library have been delayed significantly. This would negatively impact previously considered features such as: non-ChipWhisperer board compatibility, software compatibility with Windows PCs, non-public/private key encryption algorithm support, non-power analysis SCA attack support, and Higher order Power Analysis or SPA support.

Unlike the proper documentation, community support and examples that exist for many other open-source projects, such as those commonly published under the GNU General Public License (GPL), the ChipWhisperer Library is minimally documented, seemingly only developed by its creator/company (NewAE). By extension, examples for basic functionality of the ChipWhisperer are lacking, and in-line comments or documentation for existing ChipWhisperer projects and source files are nearly incomprehensible for programmers not intimately familiar with the operation of various encryption algorithms and microcontroller programming.

Regarding the assignment of an overall security score after analysis is completed, we will unfortunately be unable to implement such features, because (to our knowledge) no widely recognized certification or standard exists for protection against Power Analysis attacks, thus no numerical “scores” can be created, relative to a widely recognized standard. In addition, due to the nature of power analysis attacks, in-depth analysis of the effectiveness of circuit-level & microarchitecture-level countermeasures implemented at the hardware level and software-level countermeasures implemented in the algorithm must be considered in a case-to-case basis, similar in fashion to results reported in reputable research papers.Due to the reasons and shortcomings discussed above, we will have to re-define the scope and capability of our project to remain on schedule for project completion.

Chapter 4
Project Redesign

Figure 4 below is a screenshot of our current system’s GUI. Through the GUI, the user will be able to specify the analysis type, a sample encryption algorithm, and test board that they will use for the run. They will also be able to customize certain details of the run, such as the number of sample traces and the increment amount (CPA supports incrementally increasing traces until a successful attack).

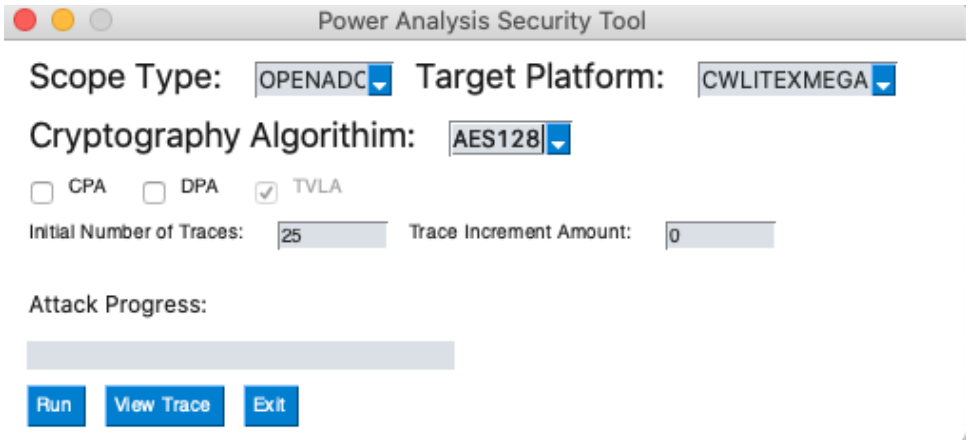


Figure 4: Our SimpleGUI program with various attack/analyzing parameters. Setup shown to run a TVLA test on an XMEGA target encrypted with AES128.

Objectives & features the project **will** still perform:

- TVLA, CPA and DPA are supported

- ChipWhisperer Target Boards with a Linux-based PC are supported
- Any Cryptographic algorithms provided by ChipWhisperer libraries will be supported

Objectives & features the project **will not** support:

- The project will not be compatible with non-Linux systems.
- Data and Results can be exported (not implemented by project deadline)
- User uploaded cryptographic algorithms will not be supported
- The system will not be compatible with non-ChipWhisperer Target boards.
- Simple Power Analysis (SPA), and Higher Order Power Analysis will not be implemented.
- Other Side Channel Analysis techniques (i.e.: fault injections, etc.) will not be considered or implemented.
- The system will require a Linux-distro PC to connect to the ChipWhisperer board for operation.
- The Results phase of the system will not assign an overall security rating to either the DUT or SUT.
- There is no guarantee that the final physical product will be “lightweight and portable”

Refined Operation

Our system is currently planned to follow a 4-step process: Setup, Collection, Analysis, and Results. During the Setup phase, the user will configure the settings that they want for their validation run. They will select the algorithm that will be tested, though currently only AES128 has been thoroughly tested (DES has limited support). They will also need to select the board type that will run their test algorithm and the type of power analysis that they wish to use. The final set of options are dependent on the analysis that will be performed. For example, for TVLA, the user would set the initial number of traces and the amount by which the trace amount is incremented by after each run. Once the user has completed the Setup step, they will press the ‘Run’ button and continue to step 2.

The Collection phase encompasses the power analysis process that is performed by the ChipWhisperer board. The system will perform the selected analysis and gather the relevant data from the resulting traces based on the analysis performed. The settings for how the analysis is performed are based on the options selected in step 1. They will control details such as the number of traces that are collected, the sampling rate, and any values necessary for the test algorithm to run. This data will consist solely of the raw trace data. Once this data has been collected, the system will move to step 3.

The Analysis phase is the point where the system will interpret the results of the collection phase and provide a rating of how secure the algorithm is regarding side channel analysis attacks. We have not been able to find an industry standard for providing a security rating such as this. As a result, we are removing the numerical rating from this step and will be solely performing the TVLA evaluation. The raw trace data from step 2 will be used in the TVLA and checked against a predetermined criterion. If the criteria are not met, the system will return to step 2 and increase the number of traces. Otherwise, it will move on to step 4.

Once the system has reached the Results phase, the user will be able to view the data that was collected as well as view the results of the evaluation. This data can be viewed within the GUI as both graphs and the raw values.

Chapter 5
Technical Specifications

This project is being built with the usage of the open source, purpose-built board called “ChipWhisperer”. NewAE (the manufacturer of ChipWhisperer) offers a range of various products in the ChipWhisperer line for specific applications and usage requirements. We chose to develop our project using the ChipWhisperer Lite hardware tool for the convenience of combining a signal capturing device, such as an oscilloscope and target. Considering this hardware is a pre-made solution, the specifications mentioned below will be brief; we cannot choose the individual components on this board. Further extension to allow for use of non-ChipWhisperer supplied targets was previously planned, but was not feasibly achievable. The software aspect of the project that we are creating is built using Python, as the ChipWhisperer APIs are also Python based [3].

Hardware & Software Components

The ChipWhisperer Lite board is designed to function as a high-quality oscilloscope with low power signal measurement capabilities. As advertised by NewAE, the ChipWhisperer can perform: *Advanced synchronous clock locking logic samples target power on related clock edges, drastically reducing sample rate requirements compared to power analysis performed with regular oscilloscopes* [4]. The board’s main components are a Xilinx S6LX9 FPGA, Atmel SAM3U microcontroller, 10-bit ADC (analog to digital converter) and a low noise amplifier [4]. *Figure 5* below is a labeled 1-part ChipWhisperer Lite board.

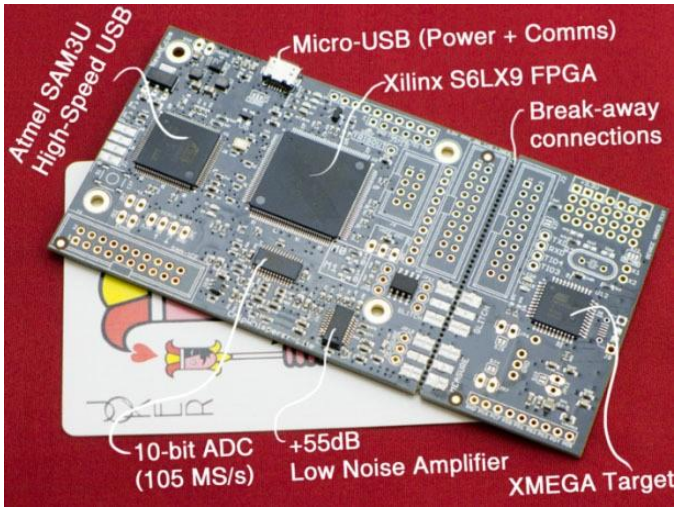


Figure 5: ChipWhisper Lite, 1-Part Board with labeled components [4]

The ChipWhisperer Lite is offered in multiple options for the included target and 1-part or 2-part boards. In our case, we decided to buy a 1-part kit that included an ARM target [5] and a 2-part kit that included a XMEGA target [6]. All ChipWhisperer Lite variants come with a target for learning and testing. The reason for buying two kits with different targets was to allow for testing of longer encryption algorithms. The XMEGA target is an 8-bit based system [7] while the ARM target is 32-bit [8]. Both 1-part and 2-part kits can have the capture and target portion

of the board separated. The 2-part board comes with the two parts already separated and comes with a 20-pin connector (for flashing target) and SMA connectors (for capturing signals). The 1-part board functions the same, but the boards are attached together on the same PCB (can be converted to 2-part if separated and have connectors soldered). *Figure 6* below is an example of the ChipWhisperer 2-part board. The choice of having a 2-part board will allow us the flexibility of attaching an oscilloscope probe to the SMA connector on the ChipWhisperer capture side of the board and test other targets (not limited to NewAE targets). Our current progress has not reached the stage of making use of these expandable abilities.

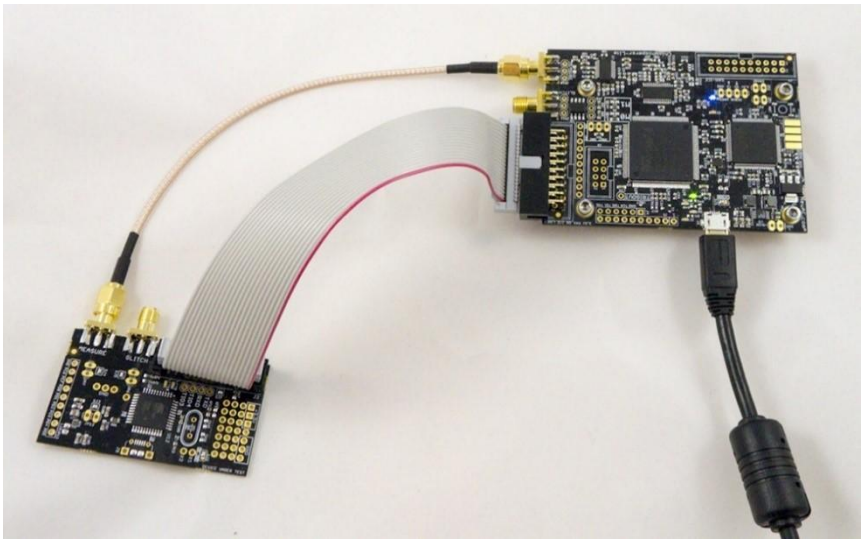


Figure 6: ChipWhisperer Lite, 2-part board, connected to target with 20-pin and SMA connectors [6]

The included ChipWhisperer software is compatible with Windows, Mac, and Linux [4]; all development we have done were either on Mac OS or Linux. The software for our project is written using the latest version of Python. Python was the language of choice considering the ChipWhisperer API is also Python based, thus this choice made logical sense. A notable aim of this project was to create a GUI for the software; the latest version of the included ChipWhisperer software does not have a regular GUI [3]. We used the PySimpleGUI package to make our GUI, as it is a relatively easy to use Python GUI development package. *Figure 3* above is the root page of the GUI of our currently developed program.

This simple GUI security validation program runs on any system with Python3 (and the program's dependencies), though we have only tested this program in Linux and Mac OS. This program is designed to obtain power traces and use these traces to compute security validation analysis parameters. The program is currently only compatible with traces from the ChipWhisperer as other capturing devices (such as a regular oscilloscope) has not been tested. In terms of compatible cryptography algorithms for analysis, only AES128 has been implanted. Implementation of other cryptography algorithms analysis is planned for development. The *Power Analysis Techniques* section below further explains the cryptographic analysis methods.

AES128 & Power Analysis Techniques

Advanced Encryption Standard (AES)

Our system will focus on the Advanced Encryption Standard (AES), a commonly used symmetric key encryption algorithm that was created to replace the relatively less secure Data Encryption Standard (DES) and triple DES (3DES) which were popular before AES's creation. AES is able to encrypt data by performing a series of rounds, during which the data is manipulated in specific ways. Each round begins with the data input, stored in a matrix of bytes, passing through an XOR gate with a portion of the secret key that is associated with that round, called the round key. The resulting value is inputted into the substitution box, or S-box, which will then output a different set of byte values. This is done by referencing a static table contained within the S-box, which relates every possible input to a unique output value. Symmetric key encryption algorithms use the S-box at the beginning of every round to further separate the secret key from the resulting ciphertext. After passing through the S-box, the data is manipulated using a series of row shifts and column mixes, which alters each byte using a function based on all of the bytes in the rows and columns, respectively. Once this is finished the data matrix will pass through the next round using a second round key made of the next portion of the secret key. The data continues through similar rounds using other portions of the secret key as round keys until the whole key has been used. The process for decrypting AES is to simply pass the ciphertext through the exact same rounds with the same round keys in reverse. *Figure 7* below depicts the encryption (left) and decryption (right) flow for a 10 round AES algorithm.

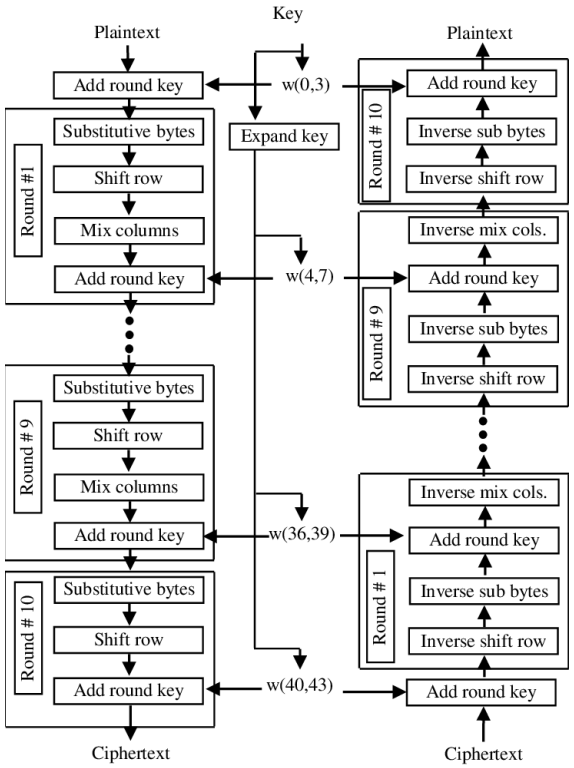


Figure 7: Block diagram of AES encryption and decryption [9]

Power analysis techniques use the power consumed during an operation using the S-box output to gain insight into the secret key. Since the time that this operation occurs is not known to the attacker, they will measure the power consumption multiple times over a certain period and use the waveforms in the analysis. *Figure 8* below shows one of these measurements, with each color representing a different power trace over a 5000 ms period.

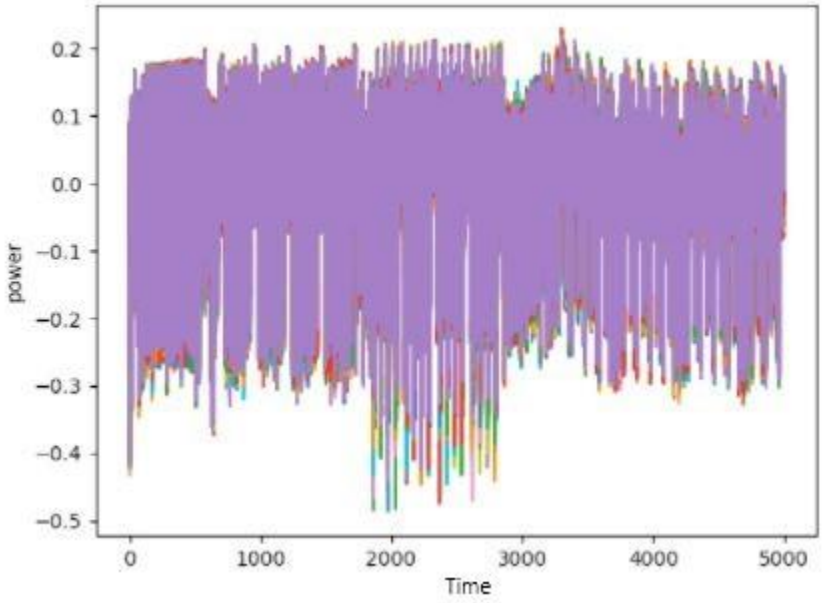


Figure 8: Graph of power consumption measures over time for all traces

Differential Power Analysis (DPA) Attack

The DPA attack uses a collection of recorded power traces for a given algorithm, and the corresponding ciphertexts per trace to produce a few best guesses at the original private key. Simply put, DPA uses the first moment for each trace to create an overall master trace, and each point of this master trace is compared to the next to find significant outliers [10]. For example, an attacker would analyze the S-box during the cryptographic computation. This is because the S-box output can show correlation between the power consumption and the plaintext and key [14]. In the case of this attacker, they would capture traces of the S-box output during the encryption of random data. The trace data can then be divided into groups according to the state of a bit (one and zero groups) in our “guessed” value predicted by the current guess at the key and the traces corresponding plaintext [15]. This technique is called the Difference of Means. The difference in means between each group allows one to deduce whether there is any significance in the proposed hypothesis [16]. I.e., a least significant bit (LSB) of 0 is compared with a LSB of 1 to prove the hypothesis (there should be a difference in power consumption if LSB output is 0 compared to LSB output of 1). The difference of means can be represented with the equation in *Figure 9.* If the averaged power trace of both groups has the largest difference from the other, it is likely that the current key guess is correct [14]. Vice versa for incorrect key guesses, incorrect group predictions will have similar averaged power traces. *Figure 10* on the next page is a block diagram of this methodology. The implementation of this can be seen in *Figure 11* on the next page, which is a single key guess from using DPA on an AES128 encryption (ARM target board). As seen in this figure, the peak mean differences of the one and

zero groups was with a key of 43. The actual key was 0x2b, which is 43 in the decimal scale. This DPA attack was successfully able to recover the single key byte (2500 sample traces were used in this run).

$$\Delta_D[j] = \frac{\sum_{i=1}^{m} D(C_i, K_n) T_i[j]}{\sum_{i=1}^{m} D(C_i, K_n)} - \frac{\sum_{i=1}^{m} (1 - D(C_i, K_n) T_i[j])}{\sum_{i=1}^{m} (1 - D(C_i, K_n))}$$

Figure 9: Equation for Difference of Means (ΔD=significant difference for each point j) [22]

In that equation above, *T* is each individual power consumption trace while *Ti* is the *i*th trace trace number. *j* represents the *j*th point in each power consumption trace. *C* represents the known inputs or outputs during an attack with *Ci* being relative to the *i*th trace. *D(Ci, Kn)* is a selection function that takes the parameters of our known input or output *Ci* along with the key guess *Kn* [22]. Where *n* is the cipher key we are using to guess (*n* would be 16 in the case of AES-128). The red boxed in this equation represents the first subcategory of traces (as sorted by the selection function) and the green boxed represents the second category of traces.

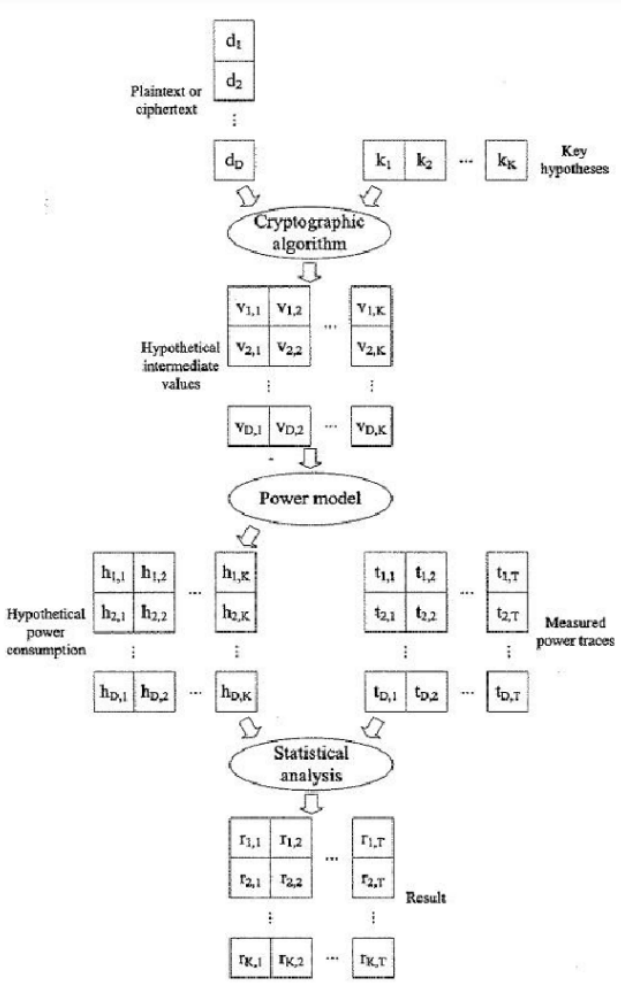


Figure 10 Block diagram of DPA attack [20]

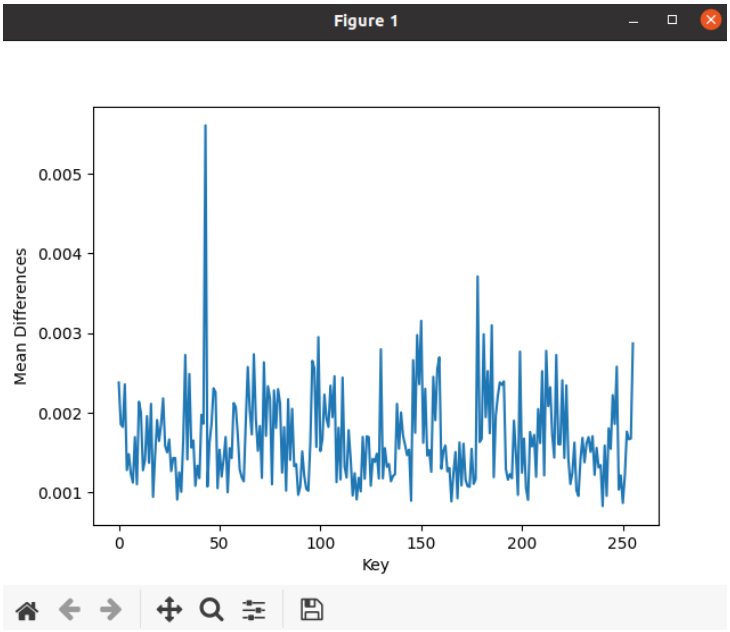


Figure 11 DPA single key guess (using mean differences). Actual key was 0x2b = 43

In our program, the DPA test that we implemented utilizes the same method as mentioned in the example. To guess all the bytes of the key, it repeats the process for guessing a single byte for the length of the key (sixteen bytes in our case). As well, to ensure maximal significance in the results, every key guess (0-255) is tested against the plain text in our implementation. Below in *Figure 12*, is the output of our program's successful DPA attack on an AES128 implementation (provided with the ChipWhisperer package). The target in this run was the ChipWhisperer XMEGA board. Variation in power consumption is very small and is not easily noticeable. Thus, in the DPA attack, a large number of traces (2000 in the example below).

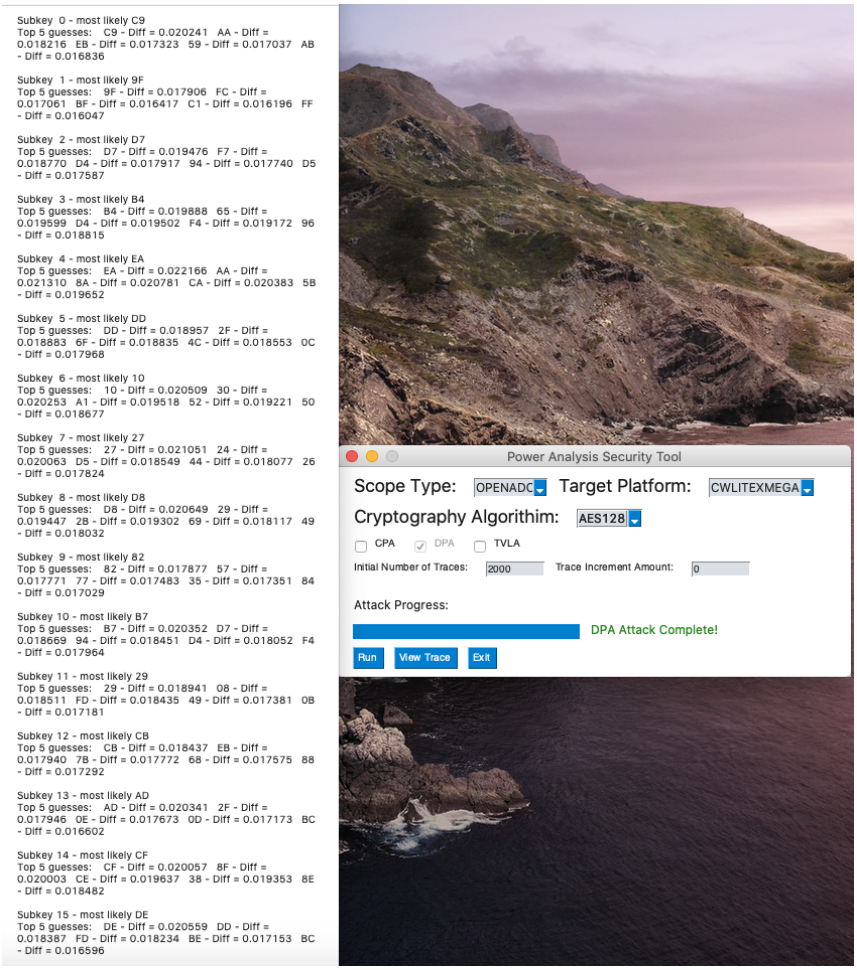


Figure 12 DPA attack results (16 byte key guesses) on XMEGA target running AES128

Correlation Power Analysis (CPA) Attack

Correlation Power Analysis seeks to gain information from the power consumption of the eight bits of data as they are passed through the S-box. The true value of these bits cannot be determined from the power consumption alone, as the rate is determined by the number of logical 1 bits in the sequence. Since only the amount of logical 1s are known, the exact sequence can be any of a large number of values [10]. For CPA to get around this, it requires a set of known data, a known s-box, and the power consumption of the s-box output. The system uses the known data and the unknown key as inputs to perform an encryption for which the power can be measured. Since the exact moment that the S-box will be used is unknown, the power trace records the power consumption waveform over a certain period of time for each byte of data from the plaintext. The system will then make a guess as to the value of the unknown key byte and determine the hypothetical s-box output values for each byte in the data using that key guess. This output value is converted into binary and each bit is summed to find the Hamming weight. The Hamming weight values are calculated for every byte using all 256 possible values for the round key. The Hamming weights are compared to each point on the power traces to find the point with the greatest correlation. It is possible that the S-box output is used multiple times in the measured time frame, leading to several high correlations for a single

guess, the greatest value is taken from among these and used as that guess' correlation value. *Figure 13* shows the formula for calculating the correlation value, where n is the sample size, x_i and y_i are the samples at point *i*, and $\bar{x}$ and $\bar{y}$ are the sample means.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Figure 13 Correlation formula for CPA

The higher the correlation value, the more likely that the guess is correct. By performing this analysis on every possible value for the key, the correct value can be determined by finding the guess with the highest correlation. *Figure 14* below is a single key guess from using CPA on an AES128 encryption (XMEGA target board). As seen in the figure, the peak correlation was with a key of 43. The actual key was 0x2b, which is 43 in the decimal scale. This CPA attack was successfully able to recover the single key byte (50 sample traces were used in this run).

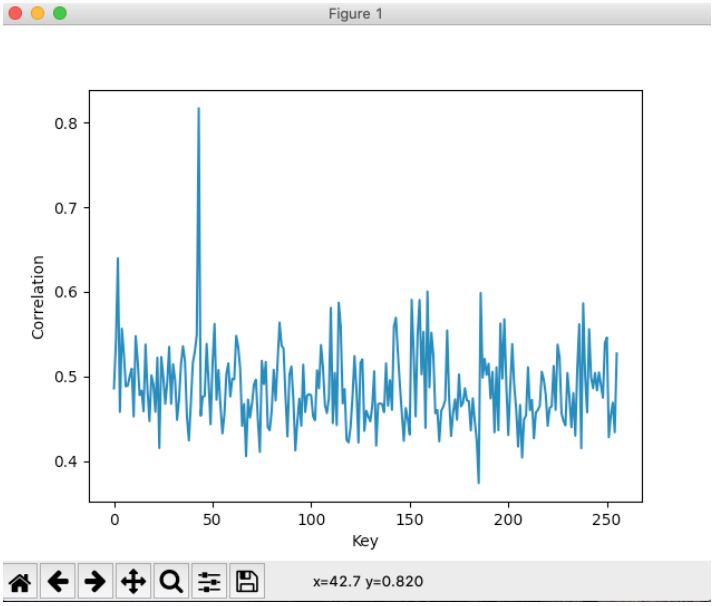


Figure 14: CPA single key guess. Actual key was 0x2b = 43

Our station will provide users with the option to perform a CPA attack on the chosen algorithm using the method described above. This feature is included so that users may test and compare different algorithms security against CPA, as well as test specific countermeasures that reduce the CPA attacks ability to find a correlation. This is intended to be used for the sole purpose of testing an algorithm's effectiveness against CPA attacks. As a result, the station will only provide raw trace data from the analysis, the resulting key guess and the correlation for each value. *Figure 15* shows the output of our station's successful CPA attack performed on the same AES128 implementation as was used for the DPA results.

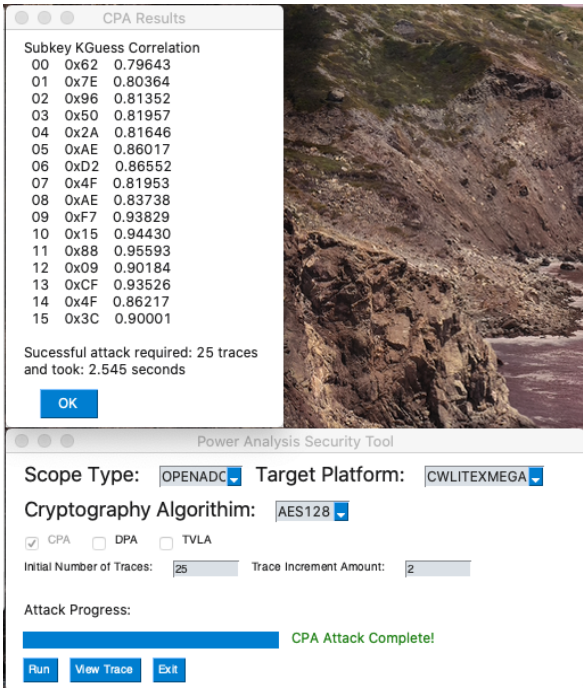


Figure 15: CPA attack results (16 byte key guesses) on XMEGA target running AES128

Test Vector Leakage Assessment (TVLA)

The TVLA considers the entire cryptographic device; both the SBS and cryptographic algorithm, as a single 'black box', where input data (such as a plaintext and private key) will produce some output data (i.e.: ciphertext) and Side Channel emissions (EM emanation, Power fluctuations, etc.). Based on the test type, recorded power traces and corresponding input data, the TVLA statistically applies Welch's t-test to determine if any side channel emission could hold information significant enough to be considered a valid "leak" [10 ,11]. By extension, this does not only examine information leaked or the weaknesses of an AES s-box for example, but evaluates the entire trace for any information that could be considered significant. TVLA can be performed either as a Specific test or Non-specific test.

Relating to real world application of TVLA, this assessment technique is also widely recognized by the U.S. National Institute of Standards and Technology (NIST), who mandates TVLA as part of its' Cryptographic Module Validation Program (CMVP); and is formally designated as security standard FIPS 140-3 [21] (Federal Information Processing Standard Publication 140-3). The standard stipulates TVLA as one avenue of assessment for conformance-style testing, though no security modules have yet been approved at the time of writing this report.

A Specific test uses two Random trace sets in a Random versus Random (RVR) comparison to find key dependent intermediate values for secret key recovery attacks, allows the cryptanalyst to compute the total Signal to Noise (SNR) from TVLA and other secret details [13, 17]. In the case of AES128, this would be the examination of its round outputs, S-box outputs, and the XOR of both input and output rounds [13, 18].

However, a Non-specific test examines a Fixed trace set to a Random trace set in a comparison called a Fixed versus Random (FVR) comparison and determines if the DUT produced a detectable leakage [12, 18]. The distribution type of the Fixed set happens to follow a random (or “Gaussian”) distribution pattern, whereas the random set follows a uniform distribution, due to the inherent nature of the measurements taken from the DUT (due to noise, silicon drift, varying temperature, external EM emissions, etc.) and its inputs all have an equally likely chance to be used.

Our implementation of 1st order Non-specific TVLA currently supports AES128 and DES for a user defined N number of traces per set.. It begins by collecting two trace sets (each containing N traces) using a specific device key (K_{dev}), generation key (K_{gen}), an initial input to encrypt N times for the fixed set (I_{fixed}), an empty 16 bytes as the initial input for the random set (I_0), and uses the previous random set’s resulting output (ciphertext) as the input (plaintext) for the next random set AES/DES encryption operation (I_{j+1}). The chosen values for this AES/DES analysis can be found in the Appendix, under section “AES/DES TVLA Dataset” and are borrowed from Rambus’ Test Requirements for AES TVLA [13]. After a set of both known and random plaintext traces are collected from the ChipWhisperer, we simply calculate the mean and standard deviation for each trace set to use in calculating the t-statistic using Welch’s t-test, as seen below (*Figure 16*). If the absolute value of this t-statistic is found to be equal or greater than 4.5, the leak is considered as valid to a confidence level of 99.999% [11].

$$t = \frac{\overline{\mathcal{F}} - \overline{\mathcal{R}}}{\sqrt{(\sigma_{\mathcal{F}}^2/N_{\mathcal{F}}) + (\sigma_{\mathcal{R}}^2/N_{\mathcal{R}})}},$$

Figure 16: Welch's t-test for TVLA [11]

As seen in *Figure 17* below, our program’s TVLA test resulted in some T-values that were above |4.5| on this AES128 encryption, generated from 5000 total traces. This means there is noticeable leakage on the encryption implementation.

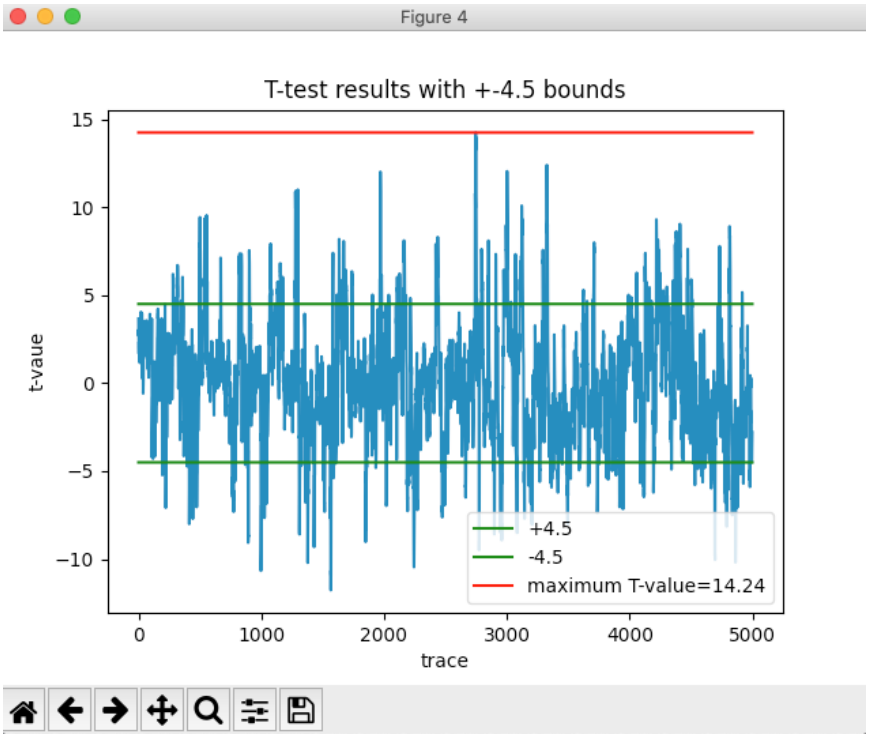


Figure 17: TVLA results for AES128 generated from 5000 traces using the CW XMEGA board.

Software Sequence Diagram

Figure 18 below is a UML sequence diagram of the operation of our project. Note that this sequence diagram has the assumption that the hardware setup is done correctly. See *High-Level System Circuit/Component Diagrams* section below for more details on this setup.

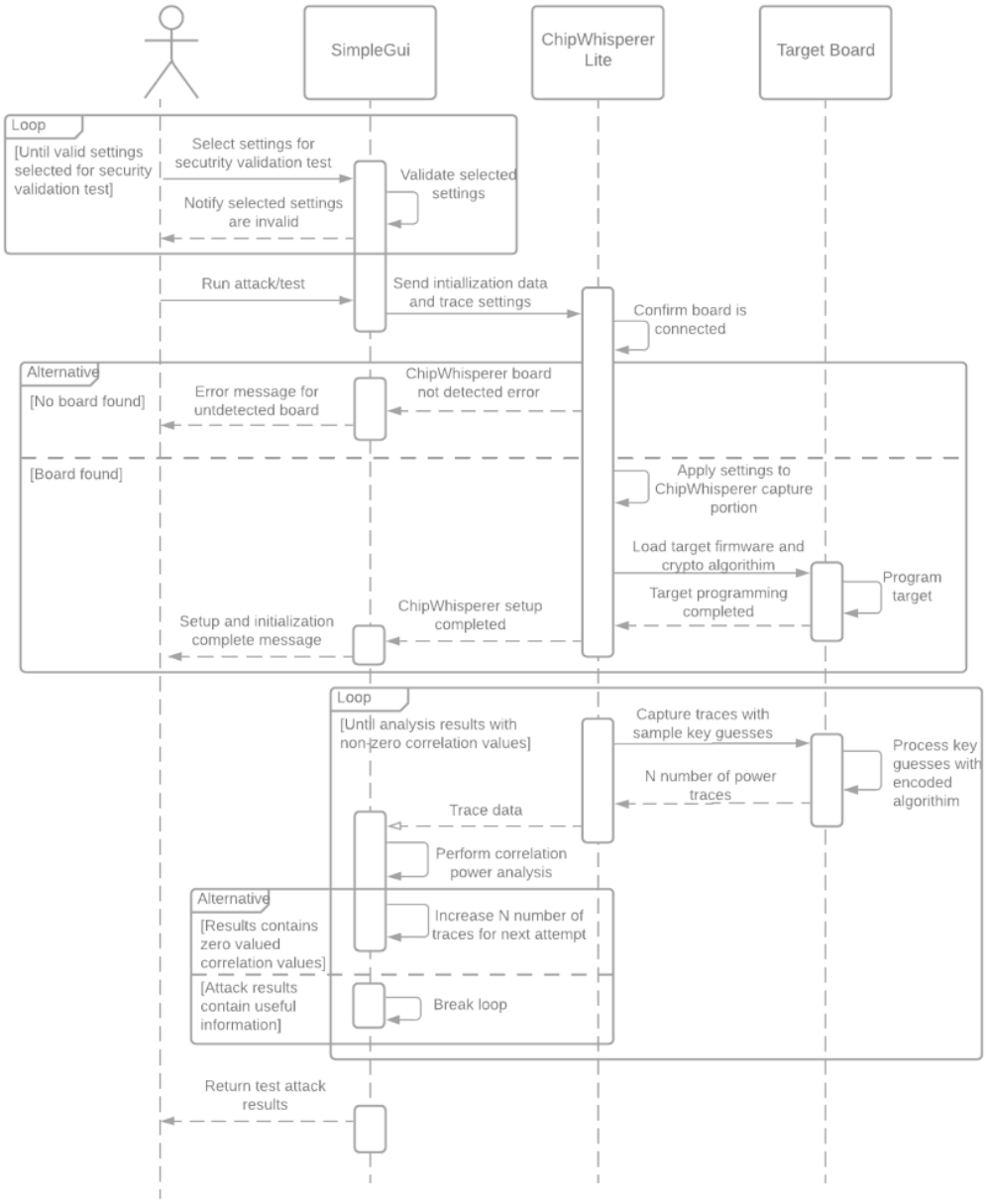


Figure 18: UML Sequence diagram of security validation project

High-Level System Circuit/Component Diagram

The high-level system circuit diagram of the hardware is simple. It consists of just the ChipWhisperer board (with target attached), micro-USB to USB A cable and a computer running a Unix based operating system (currently only tested on Mac OS and Linux). The target attached to the ChipWhisperer board in our setups are the supplied targets from the ChipWhisperer, thus no further circuitry was needed. *Figure 19* below is a diagram of this high-level system. The setup for the 2-part board is the same besides the cables that connect the target and capture portion of the ChipWhisperer Lite, as seen in *Figure 5* above.

Figure 19: ChipWhisperer Lite (1-part board) connected to laptop via USB

Full schematics of the ChipWhisperer Lite board and targets are supplied by NewAE and can be found in the link in the *Appendix.*

Internal Project Data & Testing Management

All the software that has been developed by us is stored on a GitHub repository for fluid version control among us. This repository can be found from the link in the *Appendix*. As a quick reference of this repository, it also holds tutorials provided with the included ChipWhisperer software package. The use of Git was crucial as it helped in testing and work tracking. For example, we had cases when code in the repository had bugs that did not exist in earlier versions. With the use of proper version control, we were able to backtrack changes that were made in the latest version and more easily pinpoint the cause of the bug. As well, we made use of GitHub's pull requests. If one of us were planning to make critical changes, or were adding new features, we would create a branch first. The use of branches protected the main working version of the software and isolated changes to a separate version of the code. After the branch was "done" locally, it is pushed to the repository and a pull request would be created. The pull request will ensure that another member of the team can review the changes before merging into the main version of the code. With the pull request, comments can be easily made and if there is a notable issue, an "issue" can be opened in GitHub to make this problem more visible. Though in our development progress, we have not extensively used this GitHub "issue" feature yet.

Testing of the code is always done by multiple people. First, the developer of the code/feature will perform their own multi-stages of testing. Typically (depending on code depth) will perform some unit testing, integration testing and regression testing. The unit testing will be done to ensure any new functions or sections of the code works as expected (prevents tough to catch bugs later). Then integration of the new code to the existing code is tested for compatibility, i.e., things work together as expected. Lastly the developer would do regression testing to test the whole program to make sure other parts of the program have not been

affected unexpectedly. After the code has been pushed and is under review by another member, usually regression testing is performed, and unit and integration testing is done if issues or potential issues are spotted. In essence to our pseudo-Agile software development style, whenever we are using our program, it is being tested to a degree. We are creating new requirements as we find probable improvements or features.

For project documentation (i.e., reports), we use a mix of Google Docs and Microsoft Teams for our collaborative writing. We started using Google Docs for our documentation, and later transitioned to using Microsoft Teams as it had better integration with our school accounts and it already incorporated members of our supervisor.

Chapter 6
Conclusion

Due to the hurdles that we have faced since our initial proposal and progress report, we found it necessary to re-evaluate our goals for the Security Validation Station. It was decided that the number of features would be reduced to ensure that the necessary functionality is given the proper development time that it needs. We narrowed down the design to a few integral functions and have allotted our remaining development time such that they may be fully implemented. Our primary focus is on polishing the two core functions which are fundamental to the purpose of the project. These functions are providing the user with relevant data on the algorithms' security against power analysis attacks and validating the security of an encryption algorithm using TVLA. The initial plan on allowing the user to supply and use their own encryption algorithm has been removed. This was due to our target board specific programming interface. More about this can be found in the *Reflections*.

Though our current final product is more of an SCA and TVLA demo tool, we concluded that by focusing on a smaller number of important features, it would be a more polished and functional, albeit simpler, product. Ultimately, this shift in objectives was a success as we were able to enhance the core features that we had to a more robust state. But, more importantly, we believe that this project taught us valuable topics in computer security and can be used to teach others.

Chapter 7
Reflections

When compared to our proposal, many of the objectives were not fully met, including some larger features. We planned this project with the ChipWhisperer board as our main power capture tool because of YouTube videos and online tutorials showing the ease of use of it. In our planning, we believe using the ChipWhisperer board would streamline the setup process and allow us to spend more time on developing the core features of our analyzer software. Though this was not the actual result, the ChipWhisperer board we purchased was a newer version and documentation was not flakey at best. As well, the GUI based ChipWhisperer software that we first saw in our research for this project was cancelled right before we started the project. With this, many of the documentation was not available yet. The alternative was to utilize an oscilloscope to do the capturing. Though this may have resolved issues in programming the target board with custom algorithms (we wouldn't need to interface with another board program), this option may have not been logistically possible. Considering the cost of a higher quality oscilloscope and target boards, it would have most likely required us to work in a campus lab. But this was not very feasible during the lockdown that took place. Regardless, working with the ChipWhisperer product was a learning experience in open source software and hardware.

A notable feature that was excluded from the final version of our program was the ability to use user supplied encryption algorithms. This was supposed to make our software more flexible for other than the ChipWhisperer provided encryption algorithms, since modified/different algorithms can show unique results. Our target board (supplied with ChipWhisperer Lite) used a specific programming interface that required significant modifications to any non-ChipWhisperer supplied encryption algorithms to program the board. This would take away from the ideal 'drag and drop' model that we initially planned for, and was not able to solve in our timeline. We started work on this topic relatively late in the development cycle and in hindsight, starting work on this earlier may have allowed us to complete this or realize sooner of the difficulties.

Despite setbacks to this project, we still believe that our work produced respectable results. The fact that all of us came into this project with limited experience in computer security, we were still able to grasp subjects in this complex field and perform implementations. Regarding the equipment, the usage of "rough" open source software and hardware was also new to most of us. At the end, we were able to use, modify and dissect parts of the ChipWhisperer product to allow it to work with our software. As well, development of Python based GUIs was an area that we all had minimal experience with. We were able to learn this skill and produce a presentable GUI.

Group Reflections

Though all members of the group are in the same educational program, we all brought different skills to the table. This included software, mathematical, writing, technical and communications skills that cooperatively helped us achieve our goals. Overall, we all worked

well together and valued the experience from this project. Everyone attended meetings when necessary, completed tasks on time and notified in advance if any conflicts arose. Nonetheless, we have had small conflicts in discussions of project scope, and decisions. When it came to deciding on what we wanted our project to do, our discussions were lengthy as members had different goals in mind on what we wanted our project to deliver. Similarly, with other project related decisions such as work in documentation or scheduling, we also experienced this issue. Regardless, in the end, we viewed each opinion objectively and almost always reached a collective agreement as a group on a final decision. As we reached the end of this project, from the highs and lows, we all grew together from this project.

References

[1] K. Marneweck, “The role of physical security in IoT,” Arm Community, 14-Mar-2019. [Online]. Available: https://community.arm.com/iot/b/internet-of-things/posts/the-role-of-physical-security-in-iot.

[2] L. Zhang, L. Vega and M. Taylor, "Power Side Channels in Security ICs: Hardware Countermeasures," San Diego, 2016. https://doi.org/10.1007/3-540-48059-5_25.

[3] "Software - ChipWhisperer 5.3.1 documentation," 2020. [Online]. Available: https://chipwhisperer.readthedocs.io/en/latest/getting-started.html#software.

[4] NewAE Technology Inc, “CW1173: ChipWhisperer-Lite Product Datasheet,” Mouser.ca, 13 Feb-2018. [Online]. Available: https://www.mouser.ca/datasheet/2/894/NAE-CW1173_datasheet-1859842.pdf

[5] ChipWhisperer-Lite (CW1173) Two-Part Version," 2020. [Online]. Available: http://store.newae.com/chipwhisperer-lite-cw1173-32-bit-basic-board/.

[6] ChipWhisperer-Lite (CW1173) Two-Part Version," 2020. [Online]. Available: http://store.newae.com/chipwhisperer-lite-cw1173-two-part-version/.

[7] NewAE Technology Inc., “CW303 XMEGA Target,” *NewAE Hardware Product Documentation*. [Online]. Available: https://rtfm.newae.com/Targets/CW303%20XMEGA/.

[8] NewAE Technology Inc., “CW303 Arm Target,” *NewAE Hardware Product Documentation*. [Online]. Available: https://rtfm.newae.com/Targets/CW303%20Arm/.

[9] Wadday, Ahmed & Wadi, Salim & Mohammed, Hayder & Abdullah, Ali. (2018). Study of WiMAX Based Communication Channel Effects on the Ciphered Image Using MAES Algorithm. International Journal of Applied Engineering Research. 13.

[10] M. Randolph and W. Diehl, “Power Side-Channel Attack Analysis: A Review of 20 Years of Study for the Layman,” Cryptography, vol. 4, no. 2, p. 15, May 2020 [Online]. Available: http://dx.doi.org/10.3390/cryptography4020015

[11] T. Mostaha, A. Shahverdi and T. Eisenbarth, "Silent Simon: A threshold implementation under 100 slices," in 2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), Washington, DC, IEEE, 2015, pp. 1-6, DOI: 10.1109/HST.2015.7140227.

[12] Schneider T., Moradi A. (2015) Leakage Assessment Methodology. In: Güneysu T., Handschuh H. (eds) Cryptographic Hardware and Embedded Systems -- CHES 2015. CHES 2015. Lecture Notes in Computer Science, vol 9293. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-48324-4_25

[13] Test Vector Leakage Assessment (TVLA) Derived Test Requirements (DTR) with AES. Available: https://www.rambus.com/wp-content/uploads/2015/08/TVLA-DTR-with-AES.pdf

[14] O. Gervasi *et al.*, Eds., *Computational science and its applications - ICCSA 2006: International conference, Glasgow, UK, may 8-11, 2006, proceedings, part III*, 2006th ed. Berlin, Germany: Springer, 2006.

[15] "Part 3, topic 3: DPA on firmware implementation of AES — ChipWhisperer 5.4.0 documentation," *Readthedocs.io*. https://chipwhisperer.readthedocs.io/en/latest/tutorials/courses_sca101_soln_lab%203_3%20-openadc-cwlitearm.html (accessed Feb. 25, 2021).

[16] O. Lo, W. J. Buchanan, and D. Carson, "Power analysis attacks on the AES-128 S-box using differential power analysis (DPA) and correlation power analysis (CPA)," Journal of Cyber Security Technology, vol. 1, no. 2, pp. 88–107, Sep. 2016, doi: 10.1080/23742917.2016.1231523.

[17] D. B. Roy, S. Bhasin, S. Guilley, A. Heuser, S. Patranabis, and D. Mukhopadhyay, "Leak me if you can: Does tvla reveal success rate," IACR Cryptology ePrint Archive, Tech. Rep., 2016. [Online]. Available: https://eprint.iacr.org/2016/1152

[18] Gilbert Goodwill, Benjamin Jun, Josh Jaffe, and Pankaj Rohatgi. A testing methodology for side channel resistance validation. NIST noninvasive attack testing workshop, 2011. http://csrc.nist.gov/news_events/non-invasive-attack-testing-workshop/papers/08_Goodwill.pdf.

[19] "Submission to Part II of the Walkerton Inquiry," *Professional Engineers Ontario*, April 2001, http://www.archives.gov.on.ca/en/e_records/walkerton/part2info/partieswithstanding/pdf/peo.pdf

[20] C. Wong, "Analysis of DPA and DEMA Attacks." San Jose State University Library [Online]. Available: http://dx.doi.org/10.31979/etd.9qtz-uy4r

[21] National Institute of Standards and Technology (2019) Security Requirements for Cryptographic Modules. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS PUBS) 140-3, Change Notice 3 March 22, 2019. https://doi.org/10.6028/NIST.FIPS.140-3

[22] O. Lo, W. J. Buchanan, and D. Carson, "Power analysis attacks on the AES-128 S-box using differential power analysis (DPA) and correlation power analysis (CPA)," Journal of Cyber Security Technology, vol. 1, no. 2, pp. 88–107, Sep. 2016, doi: 10.1080/23742917.2016.1231523.

Appendix

Project GitHub Repository

https://github.com/AlanLCarleton/SideChannelSecurityValidationStation

AES/DES TVLA Dataset

K_{dev} = 0x0123456789abcdef123456789abcdef0
K_{gen} = 0x123456789abcdef123456789abcde0f0
I_{fixed} = 0xda39a3ee5e6b4b0d3255bfef95601890
I_0 = 0x00000000000000000000000000000000
I_{j+1} = AES(K_{gen}, I_j) for 0 ≤ j < n

SimpleGUI TVLA Output based on NIST Workshop Welch's T-test [18]

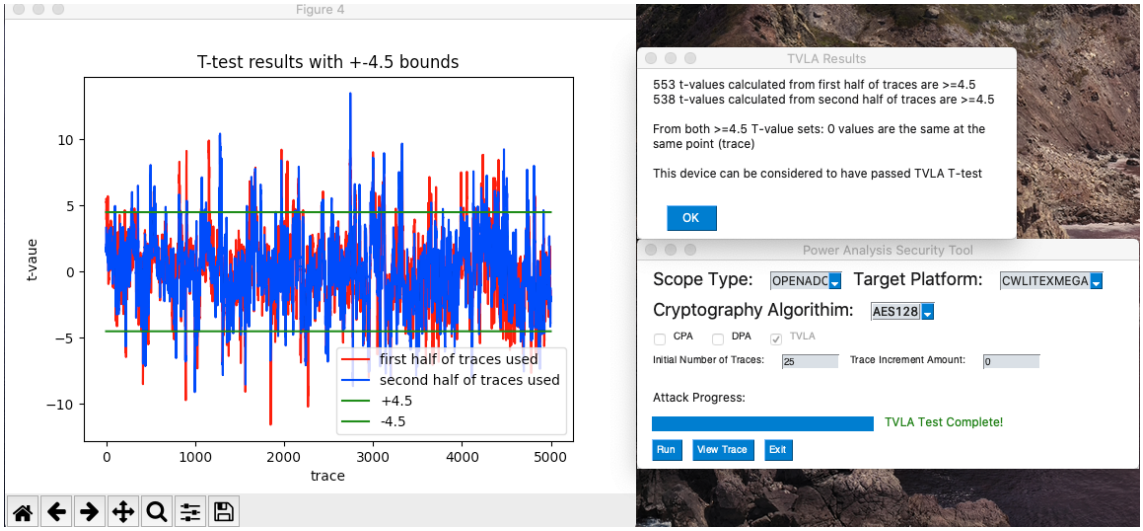


Figure 20: SimpleGUI's Moderate security TVLA results for AES128 generated using the CW XMEGA board.

In *Figure 20,* the results also show that there is leakage. Though we also implemented a moderate security level interpretation method. Our program's TVLA test resulted in a "pass" for this AES128 encryption. There are still points on the graph were the t-statistic is over |4.5|, but the criteria we used to determine the pass/failure statement was: a failure is when there is any point for which the t-test value exceeds |4.5| for both groups of t-test values [18]. Otherwise, the device passes this test. The first group of t-test values are calculated from the first half of fixed and random values, and the second group is the t-test values from the second half of fixed and random values. This method of splitting the data and running two t-tests is because each trace consists of thousands of power measurements and the likelihood of a 'false' t-test statistic exceeding |4.5| (or even a larger value) is high. These two independent t-test calculations on each half of the data and comparing the t-values at the same point will help minimize false positives [18]. Though this methodology would not pass the criteria of a moderate security level such as FIPS 140 level 3 [18].

Building an Integrated Station for Implementation Security Validation

Project Proposal

Project Members:

Alan Lin (#101036660)

Ben Bozec (#101034327)

Brannon Chan (#101045946)

Group No. 43

Project Supervisor:

Professor Mostafa Taha, Ph.D.

November 2nd, 2020

Department of Systems and Computer Engineering

SYSC4907A - Fall 2020 to Winter 2021

Proposal

1.0 Objectives:

- Each set collected by the automated power trace process will use auto-generated random private key(s) unique to each set, to provide additional authenticity to the results.
- Will utilize a NewAE ChipWhisperer-Lite.
- Will accept a variety of target boards.
- Will run DPA on a range of cryptographic algorithms.
- Require little user input as the analysis portion is largely automated.
- Final product will be portable and have a user-friendly UI.
- Automatically generated security validation reports will assign an overall ‘score’ to the Device under Test (DUT) and the cryptographic algorithm (Software under Test (SUT)) used.

2.0 Background – The State of the Art & Current Practices:

In layman terms, Power Analysis is the act of scrutinizing the input and output power of a device while it completes a cryptographic operation and extracting the secret data contained within the device using the recorded power measurements [1]. In response to this relatively new security vulnerability, new approaches to cryptography such as Elliptical Curve Cryptography (ECC) [2] provide a robust method to hiding secret encryption/decryption data.

Regarding current security practices, the U.S. National Institute of Standards and Technology (NIST), as well as the International Organization for Standardization (ISO) have published standards relating to cryptographic module(s) and their validation, such as FIPS 140-3 [3], ISO/IEC 19790:2012 [4] and ISO/IEC 24759:2017 [5].

3.0 Project Functionality – A brief description:

By applying various power analysis techniques, such as Differential Power Analysis (DPA) and Correlation Power Analysis (CPA) [1, 6], to a range of target microprocessors, microcontrollers, cryptographic algorithms and FPGAs similar to components used commonly in ‘secured’ endpoint devices, IoT and embedded applications. This project seeks to assess and report on the validity of the security implementation of such target devices.

Such target devices may be susceptible to leaking secret information via Power Analysis attacks while encrypting/decrypting data during normal operation. To report on the validity of security implementation in these devices and corresponding cryptographic algorithms, the validation station will automatically apply DPA techniques on a target device via a NewAE ChipWhisperer. Authenticity of the station’s results will be supported in part by ensuring consistent testing parameters.

The project will utilize the NewAE ChipWhisperer-Lite [6], which is an open source, purpose-built board used specifically for performing Side-Channel Analysis attacks on a wide variety of physical targets. Based on the functionality of previous industry standard platforms for

Side-Channel Attack Evaluation such as the SAKURA [7] and SASEBO [8] Projects, the ChipWhisperer toolchain provides a relatively low-cost, 'all-in-one' standardized capture tool for both education and research in side-channel attacks & specializing in power analysis attacks. By extension, the hardware aspects of this project will largely be handled by the ChipWhisperer-Lite, and the project will focus on the ChipWhisperer's application in our Integrated Security Validation Station. Seen in Figure 1: The NewAE ChipWhisperer-Lite (CW1173) Two-Part Version is the "measurement/analysis" side of the ChipWhisperer lite.

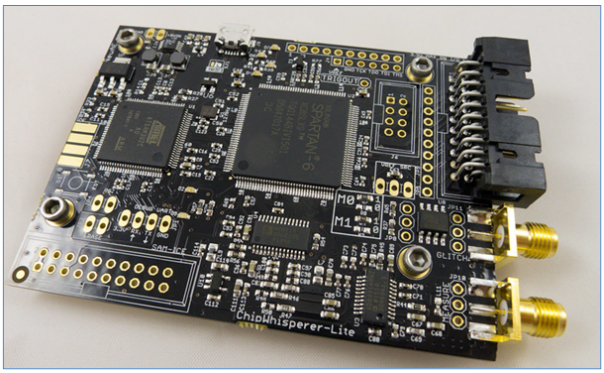


Figure 1: The NewAE ChipWhisperer-Lite (CW1173) Two-Part Version Analysis Board. Source: [9].

The finalized validation station will be largely automated, physically portable and graphically user-friendly, requiring little to no user supervision during power trace collection. Aside from setting up the physical target device, selecting the target cryptographic algorithm and configuring the automated power trace process, the cryptanalysis will be able to easily examine power trace results collected from the experiment by viewing statistically interpolated data using CPA techniques provided by robust open source Python code.

Lastly, the project will examine the overall cryptosystem (both the DUT and SUT) as a "black box" system, to determine if valid information from the black box is leaking into a side channel. This is accomplished by applying Test Vector Leakage Assessment (TVLA) [10] to the DUT; using non-specific leakage tests to compare encrypted fixed data versus unknown random data to identify valid leakages, and by extension, possible vulnerabilities in the hardware and software countermeasures implemented on the DUT. The legitimacy of possible leakages will be calculated using a statistical test, such as Welch's T-Test [10] (as seen in [11]).

$$t = \frac{\mu_a - \mu_b}{\sqrt{(\sigma_a^2/N_a) + (\sigma_b^2/N_b)}}$$

Figure 2: Welch's T-Test (from [11])

Results regarding the validity of implemented security from the DUT, gathered from the original raw power trace set, Correlation Power Analysis data and Test Vector Leakage Assessment tests will be automatically compiled into reports made available in both technically detailed and layman formats.

3.1 Additional Functionality

Given the projects' limited time frame, many features have been removed to further limit the scope of the project, and better ensure the timely completion of the core milestones listed in Section 8.0 Risks and Mitigation Strategies: of this report. However, if time allows, these features may be implemented when/where possible.

Starting with the Integrated Validation Station, it will not be implemented as a single 'all-in-one' package and will require some assembly plus a physical machine to run the stations' software. Additionally, the software will not be compatible with Oscilloscopes and will only function on Linux computers such as a Raspberry Pi, MacOS, Ubuntu, or any other Linux distribution. Regarding the analysis and validation done by both the station and software, Simple Power Analysis (SPA) [2], Higher Order Power Analysis (i.e.: second order differential power analysis) [1] will not be implemented until further notice.

4.0 Team Members & Roles:

4.1 Related Education

The three members of this project are enrolled in the same degree program, Computer Systems Engineering, which allows for flexibility on which role can be assigned to each member. It was decided that the project would be divided into three distinct components and that each member would be responsible for ensuring their component was completed on time and that it meets the standards we have set. The three components are the UI, the power analysis, and the algorithm validation. While each member is responsible for supervising their component, the tasks required to complete each component will be divided among all the group members.

4.2 Roles and Member Skills

The UI component will encompass all elements of user interaction, from selection options for the power analysis to displaying and export the results of the validation, and it will be supervised by Alan Lin. This component incorporates many of the software development techniques that we have learned throughout our program. These techniques include the use of modularity to allow for easier integration of the other components and the reduction of redundant code.

The power analysis component, supervised by Ben Bozec, refers to the utilization of the ChipWhisperer, which includes generating the initial data for the power analysis and capturing and storing the results for validation. The development of this component requires an understanding of basic cryptographic principles which we have developed in our Network and Software Security course. To properly use the ChipWhisperer within our system, we must understand how different encryption algorithms are implemented and the role of the encryption key within these algorithms. The Network and Software Security course has provided a strong foundation for researching this information.

Brannon Chan will be overseeing the algorithm validation component that is responsible for interpreting the results of the power analysis and generating a rating for the test algorithm.

To create a system that can interpret the results, our experience with data manipulation and analysis will be integral. We have gained this experience through the various courses within our program that were focused on general skills necessary to work as an engineer.

5.0 Team Collective Skills:

Our group possesses a mixture of both technical and management skills that will allow us to effectively realize the full scope of the project as we have outlined it. These skills have been learned during our shared education, through past projects, and while in the workplace during co-op placements. The primary skill that we have developed throughout our education is a proficiency in utilizing programming languages in various ways that will be useful in the implementation of the different components of our system. Through our familiarity with Python, we will be able to incorporate the Python API of the ChipWhisperer into our system to better improve the quality of our validation process. This API allows us to interact with different components of the ChipWhisperer, such as the oscilloscope and the analyzer, which will be integral to interpreting the results of the DPA.

A vital component of ensuring that the project remains functional is the use of version control software such as Git. Through our experiences in past projects both in our degree programs and in the workplace, our group members have become skilled in the use of such version control systems. By using Git, we will ensure that all changes made to our project's code are tracked, properly documented and reviewed before merging. Without the proper skill and knowledge of this system, many of the beneficial features may be unused or mistakenly used in a way that could cause damage to the existing code.

Due to the work placements that allowed some of our members to work in professional developer teams, we have been exposed to various techniques that can be used to improve different elements of our teamwork, such as communication and task management. The purpose of many of these techniques are to facilitate the documentation of each member's actions, observations and concerns throughout the project. This documentation process creates a record for members to refer to when trying to discover the cause of errors in the system or when the group is discussing tasks and concerns proceeding forward.

Throughout our years of writing programs, we have developed our understanding of proper and effective testing methods. This understanding comes from theoretical knowledge gained through our program and through exposure to the real-world testing procedures created and utilized by the professional developer teams that we worked with. Having this experience allows us to create meaningful tests for our system to ensure that we have set strict and robust standards for our system and that our system meets these standards.

6.0 Work Breakdown Structure (WBS):

1.0 Power Analysis (Ben Bozec)

- 1.1 Generate Artificial Keys
- 1.2 Perform Power Analysis

1.3 Store Input/Output Data

2.0 Algorithm Validation (Brannon Chan)

2.1 Define Standards for Algorithm Quality

2.2 Compare Analysis Output data to Standards

2.3 Generate Security Rating

3.0 UI (Alan Lin)

3.1 Option selection

3.2 Create Display for Output data

3.3 Export Data from System

4.0 Testing

4.1 Perform manual end-to-end test

4.2 Create Unit tests for individual Functions

4.3 Perform Integration tests

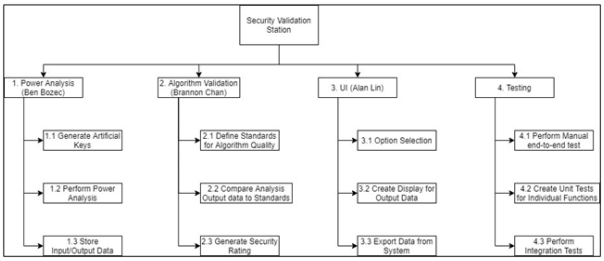


Figure 3: Work Breakdown Structure Diagram of Project Tasks

7.0 Timetable:

The estimated project progressions are formed into “phases.” Each phase objective is to divide the project into more digestible aspects of the project. The estimation of completion times below and in Figure 3: Graphical Timeline of Major Milestones are subjected to change with being ahead or behind schedule. Though those times are our target, and we plan on matching those estimates.

To summarize the project timeline, at the start, our focus is to research and gather necessary equipment. Then progress to using the hardware and software packages on test runs. After understanding the tools and equipment, we will start developing our software and continuously test. After completing the core features of our software, we will work on improving the GUI and adding “bonus” features.

Phase 1 – Research and Preparation (September 14 - November 1, 2020):

- Research
  - Side-channel power analysis theory and techniques
  - Power analysis setups and target boards
- Project Proposal Submission (Due November 2nd)
  - Rough copy due October 26th

Phase 2 – Equipment Setup and Practice (November 2 - November 15, 2020):

- Equipment Gathering and Setup Station
    - Order analysis and target boards (ChipWhisperer)
    - Install appropriate drivers and software
- Understanding Usage of Equipment
    - Read ChipWhisperer documentation
    - Follow ChipWhisperer SPA, DPA, CPA tutorials on provided targets
- Planning Infrastructure for Software Development
    - Read ChipWhisperer tools and API documentation

Phase 3 – Power Analysis Software Development (November 16 - December 6, 2020):

- Develop Barebone “Simple” Software to Run and Return Power Analysis Rating
    - Custom software to gather and present power analysis data
    - Rates the target secureness from side-channel power analysis
- Add Source and Target Boards Selection to Software
    - Software compatibility with ARM (32 bit) and XMEGA (8 bit) target boards
- Integrate Data “Beautification” to Power Analysis Results
    - Implement easy to read graphs and tables to results
- Testing of Software with the Hardware Setup (Continuously performed)

Exam and Holiday Break (December 7 - January 3, 2020 - 2021):

- Oral Presentation Form (Due December 11th)
- Prepare for Oral Presentation

Phase 4 – Hardware Setup Simplifications (January 4 - January 10, 2021):

- Oral Presentation (January 4 – January 8, TBD)
- Testing Station Simplifications
    - Clean up wiring and/or improve packaging

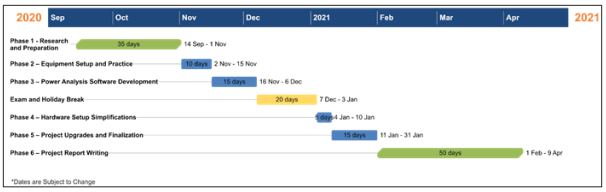
Phase 5 – Project Upgrades and Finalization (January 11 – January 31, 2021):

- “Upgrade” GUI for Software Development
    - Improve GUI user experience with visual enhancements
- Integration of External Apps (currently unspecified)
    - Integrating 3rd party table/graphing software
    - Setup emailing of results
- Progress Report (Due January 20th)
- Finalize Software and Hardware Setups (More Testing)

Phase 6 – Project Report Writing (February 1 – April 9, 2021)

- Write Final Report (Draft due February 26th)
-
- Review, Edit and Finalize Report (Due April 9th)

Figure 4: Graphical Timeline of Major Milestones



8.0 Risks and Mitigation Strategies:

With the current pandemic we face, there is a greater level of uncertainty on several topics. The biggest difference compared to other years of capstone projects is that work is predominantly (or fully) done remotely. Having limited/no direct lab resources could prevail issues from not having proper equipment to not being able to have hands-on debugging sessions as a team. To mitigate these problems, we limited the amount of specialty equipment needed and will order all other equipment promptly. Ordering the necessary equipment and receiving them on time could also raise a concern since shipping and supply chain delays are ongoing. Table 1: Risks and Mitigation Strategies below is a full list of potential risks we might run in to and possible methods to avoid them.

Table 1: Risks and Mitigation Strategies

Risks and Assumptions	Mitigation Strategies
■ Gathering all equipment in a timely manner ■ Supplier stock fluctuations ■ Shipping time uncertainties ■ Potential additional budget necessities	■ Thoroughly plan out budget to cover all equipment; arrange additional funding if expected ■ Order equipment as soon as possible ■ Select faster shipping options if budget allows
■ Potential bugs/quirks in ChipWhisperer software ■ ChipWhisperer software is open source and is in constant development ■ Proficiency in using ChipWhisperer software may take lots of time ■ Older version of ChipWhisperer software had a GUI based program [12]; latest version does not	■ Start practicing the usage of the ChipWhisperer software even before receiving the hardware ■ Look out for bug/quirk mentions in change logs and documentations ■ Run and understand the included ChipWhisperer tutorials

■ ChipWhisperer software is Python based; GUI development in Python could be challenging ■ Our team does not have experience in Python GUI development ■ Python GUI development requires understanding of additional frameworks	■ Plan our software's GUI to be 'simple' ■ Add in frills to our software after implementing the core features
■ Keeping aligned with planned timeline ■ Scope Creep ■ Adding additional functionalities (I.e. external oscilloscope compatibility) ■ Software GUI "bells and whistles" ■ Unexpected software and hardware technical difficulties ■ Schoolwork overload	■ Ensure detailed project deliverables and goals are specified; only extend scope if needed to unblock progress or core features are completed ■ Thoroughly read software and hardware documentation ■ Plan for expected heavy schoolwork weeks
■ Remote work limitations ■ Sharing ChipWhisperer boards (budget dependent) ■ Remotely debugging equipment issues as a team ■ Potential difference in home lab environment setups	■ Purchasing a ChipWhisperer board for each team member (if budget allows) ■ If each team member cannot have a board, ensure members that focus on directly using the board (I.e. not working on GUI) gets enough time with it ■ Make extensive use of equipment documentation and online resources ■ Utilize a virtual machine to create dedicated work environment ■ Document details/requirements for home lab setup

9.0 Component and Facility Requirements:

Side-channel power analysis typically utilizes an oscilloscope for capturing data from targets. We decided to use the ChipWhisperer Lite board for our project for several reasons. The ChipWhisperer board has a capture portion that functions as an oscilloscope [6]. This capture portion of the ChipWhisperer can measure small signals, which is necessary for power analysis. The ChipWhisperer Lite also includes a target, which reduces the need for us to

source target boards at the start. The entire ChipWhisperer product (hardware, software and firmware) is open source, which will open additional flexibility in our usage of the product.

The alternative was to use a regular oscilloscope, but this option would not be as feasible since we have very limited access to the school's lab for oscilloscope use. The idea of purchasing more than one decent oscilloscope and some target boards for individual home use would farther put our team out of the allotted budget. Thus, the ChipWhisperer boards made sense for our situation. Table 2: Project Required Components below is a breakdown of the core equipment our project will require.

Table 2: Project Required Components

tem	Description	Costs (*see notes below)
ChipWhisperer-Lite CW1173) 32-bit Basic Board	NewAE Technology open-source ardware, software, and firmware tool package for testing embedded security. One part of the board is the capture portion and other part is a STM32F3 arget (ARM).	x $250 USD ($328.89 CAD) *
ChipWhisperer-Lite CW1173) Two-Part Version	ChipWhisperer-Lite board that has the arget pre-separated from the capture portion. Included are the necessary connectors and wires for easily connecting to other target boards.	x $325 USD ($427.56 CAD) *
Desktop/Laptop Computer	Must be Linux based or has a Linux virtual machine installed.	N/A

*Canadian currency conversion performed on October 26, 2020 by Google's currency converter
*Shipping costs and taxes are not included in the "Costs" prices

Though most of our work is planned to be done remotely, as a team, we do plan on meeting together to work on tasks that cannot be done efficiently remotely. We would contact Carleton department facilitators to safely arrange lab time.

10.0 References

1] J. Zhang, L. Vega and M. Taylor, "Power Side Channels in Security ICs: Hardware Countermeasures," San Diego, 2016. https://doi.org/10.1007/3-540-48059-5_25.

2] J.-S. Coron, "Resistance Against Differential Power Analysis For Elliptic Curve Cryptosystems," in *Lecture Notes in Computer Science*, vol. 1717, Springer, Berlin, Heidelberg, 1999. https://doi.org/10.1007/3-540-48059-5_25.

3] Security Requirements For Cryptographic Modules," Gaithersburg, MD, 2019. https://doi.org/10.6028/NIST.FIPS.140-3.

4] *nformation technology — Security techniques — Security requirements for cryptographic modules,* ISO Standard ISO/IEC 19790:2012 , 2012.

5] *nformation technology — Security techniques — Test requirements for cryptographic modules,* ISO Standard ISO/IEC 24759:2017, 2017.

6] C. O'Flynn and Z. Chen, "ChipWhisperer: An Open-Source Platform for Hardware Embedded Security Research," in *Constructive Side-Channel Analysis and Secure Design. COSADE 2014. Lecture Notes in Computer Science*, vol. 8622, Halifax, Canada, Springer, Cham, 2014. https://doi.org/10.1007/978-3-319-10175-0_17.

7] Y. Okuyama, "SAKURA Hardware Security Project," 2016. [Online]. Available: http://satoh.cs.uec.ac.jp/SAKURA/index.html.

8] A. Satoh, "Side-channel Attack Standard Evaluation Board (SASEBO)," 2012. [Online]. Available: http://satoh.cs.uec.ac.jp/SASEBO/en/board/index.html.

9] ChipWhisperer-Lite (CW1173) Two-Part Version," 2020. [Online]. Available: http://store.newae.com/chipwhisperer-lite-cw1173-two-part-version/.

10] M. Randolph and W. Diehl, "Power Side-Channel Attack Analysis: A Review of 20 Years of Study for the Layman," Blacksburg, VA, 2020, 4, 15.

11] T. Mostaha, A. Shahverdi and T. Eisenbarth, "Silent Simon: A threshold implementation under 100 slices," in *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, Washington, DC, IEEE, 2015, pp. 1-6, DOI: 10.1109/HST.2015.7140227.

12] Change Log - ChipWhisperer 5.3.1 documentation," 2020. [Online]. Available: https://chipwhisperer.readthedocs.io/en/latest/changes.html.

OBJ