Alec Austin

Meghana Gathpa

Joshua Hardy

Rachana Kandadi

Kalyani Padaraju

**Project Final Report**

**1. Abstract**

Side-channel analysis is a well known method of uncovering cryptographic secrets through analyzing hardware. Many methods have been created to defend against side-channel analysis. One such method is to divide the cryptographic algorithm between software and hardware. Based on this method, we created a hardware-software implementation of the cryptographic algorithm 3DES. Our python scripted software implementation is collaborated with the Xilinx PYNQ-Z2 board. The implemented 3DES can only accept ASCII inputs and is not in the traditional 3DES form, but it is otherwise a successful implementation.

**2. Introduction**

Side-channel analysis is both a well known and very powerful technique for uncovering cryptographic secrets. It involves analyzing leaked information from a machine running a cryptographic algorithm in order to gain information about the key being used for the algorithm. These leaks can include things such as power consumption, time being taken, the program counter, and others, and the leaks themselves are known as “side-channels.” Side-channel analysis has led people to question the true security of cryptographic algorithms, and as a result, people have looked into ways to protect cryptographic algorithms against side-channel analysis attacks. While there are many methods for defending against side-channel analysis attacks, our focus was on the idea of splitting a cryptographic algorithm between the hardware and software. The design of the triple DES as an encrypt-decrypt-encrypt for encryption and decrypt-encrypt-decrypt for decryption allows to split the implementation between hardware and software. This can be done by using a single DES implementation with different keys i.e K1,K2,K3 for the encryption-decryption process.

Based on this idea, we created a version of 3DES that splits its DES rounds between a hardware and a software implementation of DES.

**3. Related Work**

3.1. Research Work

**Security and Privacy Challenges in Cyber-Physical Systems**

This paper focuses on the CPS security and security-performance trade-off. Ensuring secure communication and preventing untruthful data from spreading across a system is a critical concern because, as noted above, the existing protection techniques do not suit the particularity of this kind of system.

Safety and Security Objectives in CPS:

1. Confidentiality is the ability to prevent information and data from being exposed to any unauthorized individual or party from inside or outside the system.

2. Integrity is the ability to keep data as it is and prevent any unauthorized manipulation.

3. Availability – the systems’ ability to provide services and output products in a timely manner.

4. Authenticity: the ability to guarantee that all parties participating in any CPS processes are supposed to do so.

5. Robustness is the degree to which CPS can continue to work properly, even in the presence of limited disturbances.

6. Trustworthiness is the degree to which people can rely on the CPS to perform required tasks under specific domain constraints and according to specific time conditions.

The Side Channel Attack (SCA) exploits leaked side channel information including energy consumption and execution time from the chips in the hardware implemented cryptosystems. The goal of the SCA is to reveal the secret keys and they can be applied to many running cryptographic devices. The Differential Power Analysis attack (DPA), is a version that utilizes the inconstancy of power consumption to detect the secret information using statistical techniques. In this attack there are two main techniques: data collection and data processing. The Comparative Power Analysis (CPA) assumes that an attacker can input user-defined message to RSA device, and can reveal the secret key by less power consumption traces than that required by DPA.

One good solution to avoid these attacks is to propose efficient and low power cryptographic implementations for the encryption algorithms.

CPSs face several critical challenges, including information security, privacy-related concerns and the tradeoff between security and performance (AlDosari, 2017).

**Security in IoT**

This paper gives a brief overview of security issues in iot devices at different layers of iot devices and their countermeasures. As we know Iot devices are the things embedded with sensors, actuators, software that are connected via the internet which can be controlled or operated from any place anytime. There are a lot of iot devices in the market without proper security and privacy that leads to leakage of user information or prone to security attacks.

The iot developers cannot provide security to the level of standard internet because of few limitations of iot devices. Most of the iot devices are operated with rechargeable batteries and iot devices require less computation .In order to apply security algorithms on iot devices they require resources and few algorithms take large computation time. So, in order to overcome these limitations they came up with a few algorithms like “identity based encryption schema”,“encrypted query processing algorithm” in which the iot data is encrypted using lightweight cryptography algorithm and stored in the cloud and allows efficient database query processing on the data stored in the cloud.

The attacks in the iot devices are classified based on iot structure such as physical attacks, software attacks,network attacks, encryption attacks . Most of these attacks can be prevented by performing encryption, authentication and access control mechanisms and also by maintaining the data integrity. There are few authentication and access control schemes like identity based authentication schema where the devices are authenticated using gateways which are connected controllers which are connected centralized data. They had proposed a new authentication process using a unique device fingerprint which is different from conventional authentication schemas. And explained countermeasures for smart lock vulnerability by touch based communication with the lock by using a wearable device. They had proposed an architecture model for secure communication by introducing a new device called iot security service provider. This device monitors the certificates, authentication and session establishments.

In this paper they discussed layerwise security problems and their solutions. The Iot perception layer communicates with the physical world to exchange and collect the data.sometimes there is a possibility of faulty nodes, these can be prevented by faulty detection algorithms and intrusion detection models (Yang, Wu, Yin, Li, & Zhao, 2017).

**Scheduling with Privacy Concerns**

Resources shared between processes lead to side channels that can leak information from one process to another. In a side channel, one process tries to learn something about the operation of another without the latter’s cooperation. Side channels focus on information that is leaked incidentally by a victim process.

Attacks motivate the design of scheduling policies wherein a malicious attacker cannot learn the traffic pattern of the other users of the systems reliably. The policy design of the scheduler involves deciding which user’s job (from the queued jobs) to serve upon completion of the current job. Some of the commonly used schedulers are first-come-first-served (FCFS), round-robin (RR), shortest-job-first (SJF) and priority schedulers. Performance of a scheduler is measured in one of several metrics including, throughput (number of job completions per unit time), average delay (the difference between the job completion time and the job arrival time), fairness (a metric to measure if the resource of interest is being distributed equally/ fairly between the processes) et. A scheduler often had to make a calculated trade-off among these conflicting metrics. However, in a multi-tasking environment where jobs from multiple non-trusting users are served, one should also take into account the privacy that a scheduling policy offers. One of the most commonly deployed scheduling policies is the FCFS policy. The FCFS policy is simple to implement and guarantees small delay to the users. The policy, however, provides excessive information leakage to an attacker by virtue of providing high correlation between the legitimate user’s request pattern and the delays experienced by the attacker.

The system model is a scheduler used by two users, Alice and Bob. The scheduler serves jobs at the rate of 1 job per unit time. Bob is assumed to know accurately the time when his jobs are issued, and their corresponding completion times. Knowing the delays experienced by his jobs, Bob uses this information to guess the arrival pattern of jobs from Alice. Alice’s arrival process is modeled as a Poisson process of rate ƛ\_2, with all the jobs of unit size. Bob is assumed to know the value of ƛ\_2. Bob is free to choose when to issue his (unit sized) jobs. Bob is interested in learning reliably how many of Alice’s jobs arrived during fixed intervals of time defined by periodically spaced clock ticks of duration c.

If the scheduler uses a Time-Division-Multiple-Access (TMDA) policy, it allocates dedicated time slots to process jobs from each user. As a result of this reservation of time slots for users, the arrival pattern of one user does not influence the completion times of jobs of the other, and they are therefore statistically independent of each other. Therefore, of all scheduling policies, TDMA results in the heist estimation error for the attacker. Though from a privacy perspective TDMA is an optimal scheduling policy, by reserving slots for users, TDMA loses out on benefits of statistical multiplexing, and is neither throughput, nor delay optimal.

Delay plays a crucial role in several applications, e.g. real-time tasks running on a processor. It is therefore important for a scheduler to introduce as minimal delay as possible. The delay metric is crucial in the ‘normal’ mode of operation of the scheduler. In this paper, the delay offered by a scheduling policy is defined as the average delay incurred by a job (averaged across all users of the scheduler) when multiple users issue jobs according to Poisson processes.

A scheduling policy that has been optimized to perform best on one of the metrics is most likely not the best in another. TDMA is the best policy with respect to privacy metrics. However, it uses a fixed resource allocation scheme which is sometimes not practical. This is because user’s job patterns are rarely periodic and would therefore ensure server delays. And also, the policy places a hard limit on each user’s available processor time to half the total available (in the case of two users), resulting in poor resource utilization. The TDMA is therefore suboptimal both on the metrics of throughput and delay. The sub-optimality arises as a result of the scheduler taking vacations, i.e., TDMA allows for the scheduler to stay idle even when there are unserved jobs awaiting in the queue to be served. Schedulers which do not allow for this are said to be non-idling (IEEE Information Theory Society, & Institute of Electrical and Electronics Engineers, 2012).

**A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks**

The paper proposed a framework for analysis of cryptographic implementation that includes a theoretical model and an application methodology. From a theoretical point of view, it shows the formal connections between metrics and discusses their intuitive meaning. From a practical point of view, the model implies a unified methodology for the analysis of side-channel key recovery attacks. The metrics like the success rate of the Adversary and the guessing entropy are discussed. The information theoretic metrics like conditional entropy and Shannon's conditional entropy are discussed along with the limitations. The relation between the evaluation metrics are discussed like Asymptotic meaning and the comparative meanings of conditional entropy. An evaluation methodology for side-channel attacks intends to analyze both the quality of an implementation and the strength of an adversary, involving these five steps:

1. Define the implementation
2. Define the target
3. Evaluate the information
4. Define the Adversary
5. Evaluate the security.

Additional to these metrics, it is often interesting to define a Signal-to-Noise Ratio, since noise insertion is a generic countermeasure to improve resistance against side-channel attacks, it can be used to plot the information theoretic and security metrics with its respect (Standaert, Malkin, & Yung, 2009).

**Machine learning for SCA**

In side-channel attacks where the model is poorly known, machine learning algorithms can be used. In this paper, the authors conduct experiments of different machine learning techniques, each of them tuned and trained differently, showing the importance of proper tuning and training. The paper proves that in unrestricted but practical scenarios, machine learning techniques can be more powerful than the template attacks. The template attacks typically use Bayes Theorem, where the time samples are assumed not to be independent. A new measure called the Data Confusion Factor is used to analyze the results of different techniques used. Side-channel attacks on block ciphers are considered for which divide and conquer approach can be used.

The data used to perform experiments is the publicly available traces of DPA contest v4. These traces are extracted while executing a low-cost masking protection on AES. 20000 random measurements that are selected uniformly are used for each scenario. The original traces have a high signal to noise ratio, so independent gaussian noise is added to investigate different scenarios. In the first case study, the data is classified into 16 different uniformly distributed classes. It is observed that for low noise, each class is completely separable whereas for high noise, the classes totally overlap. That also means the correlation between time instances is high for low noise and the traces are nearly uncorrelated in case of high noise. The second case study targets on 8-bit intermediate variable resulting in 9 binomially distributed classes. In this case, the correlation is slightly higher for low noise and lower for high noise.

The machine learning algorithms used are Support Vector Machines, Random Forest, Rotation Forest and MultiBoost algorithm. The datasets are divided as 2/3 for training and 1/3 for testing. 10-fold cross-validation is conducted considering all the parameters and averaged results of individual folds are noted. For parameter tuning, the best results are obtained using support vector machines. This might vary with the parameters chosen. This method is only recommended when there is enough time for extensive tuning. Otherwise, Rotation Forest can be used. Overall, in testing the SVM algorithm performed better with accuracy of 91.1% in comparison to template attack with accuracy 73.44% for the same scenario. However, when the noise is increased, the SVM becomes the worst performing algorithm for the binomial distributed hamming weight classes. An important observation is that the accuracy of 16 uniformly distributed classes is higher than 9 binomially distributed. The results from Data Confusion Factor also proves that ML techniques perform better when compared to the template attacks (Picek, Heuser, Jovic, Ludwig, Guilley, Jakobovic, & Mentens, 2017).

**Trojan Side-Channels: Lightweight Hardware Trojans through Side-Channel Engineering**

Due to the recent fabless trends in the semiconductor industry, malicious hardware components such as “Hardware Trojans” can be covertly inserted at the foundry to implement hidden backdoors for unauthorized exposure of secret information. This paper proposes a new class of hardware Trojans which intentionally induce physical side-channels to convey secret information.

There are 3 general approaches mentioned to detect the hardware Trojans which are hardly feasible. The paper presents two different approaches for TSCs on Xilinx FPGAs that are built upon spread spectrum theory and artificial leakage functions induced on the key schedule of cryptographic algorithms.

TSCs based on spread spectrum communications leak the information contained in a single bit along with many code bits which change much faster than information bits. It uses a pseudo random code generator to create code sequences. An XOR operation is performed on the sequence and is forwarded to the Leakage Circuit (LC). This sets up a secret CDMA channel in the power side-channel. The attacker performs a correlation demodulation on measurement points of subsequent clock cycles.

In the second approach, the TSC leaks the secret information obtained during the run of a block-cipher’s key schedule. The TSC proposed demonstrates an attack on AES 128 block cipher and uses an artificial intermediate state which leaks a single bit. If at least a one byte leakage is successful, it becomes less difficult for an attacker to mount a brute force from the reduced key space (Lin, Kasper, Güneysu, Paar, & Burleson, n.d.).

**Lost Cost Solutions**  
 The paper seeks to provide a generic solution to protect cryptographic algorithms from simple side channel attacks without adding a significant amount of execution time. This is done by adding dummy instructions to blocks of code until all blocks of code are side-channel equivalent. This requires a careful selection of a side-channel atomic block for all of the code blocks to match.  
 The method starts by identifying a common side-channel atomic block for all processes. The processes are then written as a repetition of that common atomic block. A bit is used to keep track of when a process has no more atomic blocks to execute, which allows the process to be chained together. A counter is also maintained to keep track of which process is currently running.

This process has been shown to work for RSA using both the sliding window version and simplified versions. It also has been shown to work for two different types of elliptic curve cryptography, one over a large prime field and one over a binary field. As such, this method should allow a large amount of cryptographic algorithms to be made secure against simple side-channel analysis. Furthermore, this method can be combined with methods to prevent more sophisticated analysis (Chevallier-Mames, Ciet, & Joye, 2004).

**Authenticated Encryption for stopping SCA**

This paper is about AES based authentication encryption schema (ALE-authenticated lightweight encryption) for protecting FPGA bitstreams from the side channel attacks and physical attacks. As we know, FPGA bitstreams are rearranged based on the requirement of application. In order to protect the bitstreams from eavesdropping and side channel attacks they implemented authenticated encryption with threshold masking on FPGA bitstreams.

Here they implemented ALE schema along with the other encryption authenticated schemas like AES-OCB,AES-CCM and AES-GCM. Initially, all these authenticated encryption schemes are implemented unprotected over three different technologies. Among all these five authenticated encryption schemas ALE(authenticated lightweight encryption) has recorded best performance and requires three times less resources when compared to others. Here, the authenticated encryption schemes are still vulnerable to some of the side channel attacks such as differential power analysis attack and physical attack. In order to protect the AE schemas from SCA they introduced masking as countermeasures. They have implemented ALE using threshold masking technique which is based on secret key sharing and multiparty computation.

All these five protected schemas are implemented on three technologies and from the results it is concluded that ALE is efficient in terms of performance and resource utilization when compared to others. When we compare the gate count values for both protected and unprotected implementation, they are almost similar. Here, we can understand that ALE provides authenticated encryption on low-cost FPGA also protects from physical and side channel attacks (Bogdanov, Moradi, & Yalcin, 2012).

**Program Counter Security**  
 It is important for cryptographic algorithms to be resistant to both standard cryptographic attacks and side-channel attacks. Having the hardware and software work together can help prevent side-channel attacks, and if the hardware can guarantee that it will only leak the program counter, then software can be designed or transformed so that the algorithm will be secure against side channel attacks. What is being leaked is known as a transcript, and the guarantee is that the program counter will be the only thing the transcript. Meeting this guarantee allows protection against timing attacks and simple power analysis, and the requirements of this guarantee align with common side-channel prevention techniques.

Any code that is secure if the hardware only leaks the program counter is known as PC-secure, and any code that has no key-dependent branches is PC-secure. There are also tools that can be used to determine if a program is PC-secure, and tools that can transform a PC-insecure program into a PC-secure program. Libraries that restrict control flow as well as tools that double check compiled code can also be used to help write a PC-secure program, although there will be a degrade in performance as a result of guaranteeing PC-security.

There are many areas of research for protecting cryptographic algorithms against side-channel analysis, but they are all very primitive and require much further research to successfully secure a cryptographic algorithm. Even PC-secure algorithms need to be improved to protect against more types of side-channel attacks as well as to determine the best contract between hardware and software to guarantee what the hardware is allowed to leak while keeping the software’s algorithm secure (Molnar, Piotrowsi, Schultz, & Wagner, 2006).

**Cache-based SCA Defense**

Software-based side channel attacks exploit information related to cache and branch predictors. There have been software-based solutions to defend cache-based attacks. In order to attain high level security, many such techniques are combined which results in performance overhead. To avoid that, a couple of hardware solutions to redesign caches have been proposed. These include Partition-Locked cache (PLcache) and Random-Permutation cache (RPcache). In this paper, the authors propose three new approaches to defend against both access-driven and timing-driven cache attacks. In access driven attacks, the attacker has control over one or more spy processes that share the cache with the victim process. In timing driven attacks, the attacker sends multiple requests to target crypto process and evaluate execution time and record cache hits and misses that can be used to derive the key. In this paper, the ARS algorithm is used to demonstrate the attacks and countermeasures.

The first proposed approach is preloading to protect PL cache. The basic idea of preloading is to load all the critical data into the cache right before crypto operations begin. Preloading itself doesn’t provide enough security since the attacker can still manipulate the cache state after the preloading process. Preloading implemented along with PL cache can protect against access and time driven attacks. The key idea is to preload critical data into PL cache, so that any attempt to access critical data results in cache hit and defeats time-driven attacks. The disadvantage of this approach is when all the critical data is too big to fit into the cache.

The second proposed approach is to secure RP cache with information loads. The previously proposed RP itself defeats access-driven attacks (by randomizing cache line mapping) but not time-driven attacks. The information loads are a set of instructions that inform the software when the critical data is not in the cache (a cache miss). There are several different ways to implement information loads. The authors of this paper chose a low-overhead cache-miss trap because it has low hardware complexities.

The third proposed solution is securing regular caches with information loads. The difference between Random-Permutation cache approach and this one is that permutations are changed only when it is necessary i.e. when there is a cache miss of the critical data. The results of all the experiments and their comparison with traditional methods are presented in the paper. From the results presented in the paper, it can be concluded that the performance overhead is much lesser in the integrated approaches proposed compared to the software only implementations. And the security or efficiency is much better when compared to the hardware only approaches (Kong, Aciiçmez, Seifert, & Zhou, 2009).

3.2. Research - Contribution

**Security and Privacy Challenges in Cyber-Physical Systems**

This paper identifies three kinds of available resources that need to be regulated: physical, computational, and networking. This paper gives a detailed description of Cyber-Physical Systems (CPSs), their challenges (including cyber-security attacks), characteristics, and related technologies. There is also a focus on the tradeoff between security and performance in CPSs. There is a presentation on the most common Side Channel Attacks on the implementations of cryptographic algorithms (AES and RSA) with the respective countermeasures against these attacks (AlDosari, 2017).

**Security and IoT**

This paper gives a brief understanding of iot devices and their security concerns. Explains different types of attacks in iot devices. Discuss about different authentication and access controls protocols and mechanisms. Explained countermeasures for smart lock vulnerability and physical and network attacks. Discussed data integrity and authentication challenges and improper functioning of Imd devices (Yang, Wu, Yin, Li, & Zhao, 2017).

**Scheduling with Privacy Concerns**

This paper begins with an explanation as to how a timing side channel is created. From there the information leakage of a two-user system is studied. After an introduction of a measure of privacy, a demonstration is provided that no schedule can provide maximum privacy without idling. As a result of this, it is concluded that no policy can simultaneously be delay and privacy optimal (IEEE Information Theory Society, & Institute of Electrical and Electronics Engineers, 2012).

**A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks**

The paper came up with a unified framework approach which is centered around a theoretical model. It evaluates the leakage functions with the security and information metrics. The paper aimed to bridge the understanding of physically observable cryptography to the exploitation of actual leakages in experimental key recoveries. It typically determines the basics to compare the side-channel adversaries.(Standaert, Malkin, & Yung, 2009).

**Machine learning for SCA**

This paper identifies the vulnerabilities of template attacks and shows that machine learning techniques can be more successful than template attacks. The emphasis is on the importance of proper algorithm selection and parameter tuning phase for ML techniques. At the time a new measure called Data Confusion Factor is used to evaluate the success of the implemented algorithms (Picek, Heuser, Jovic, Ludwig, Guilley, Jakobovic, & Mentens, 2017).

**Trojan Side-Channels: Lightweight Hardware Trojans through Side-Channel Engineering**

Considering the serious threat that can be caused by the Hardware Trojans, the paper introduced a new class of hardware trojans that are intentionally induced on Xilinx FPGAs to create side-channels. The paper evaluates the feasibility of TSCs on the Xilinx FPGAs. It provides the simple implementation methods and the detectability of the Trojan Side-Channels (Lin, Kasper, Güneysu, Paar, & Burleson, n.d.).

**Lost Cost Solutions**  
 This paper provides a generic solution for protecting algorithms against simple side-channel analysis attacks. This method is also capable of being paired with other methods that can defend against more complex side-channel analysis attacks (Chevallier-Mames, Ciet, & Joye, 2004).

**Authenticated Encryption for stopping SCA**

This paper focuses on Aes based authenticated lightweight encryption algorithm for fpga bitstreams. ALE is tested on three different technologies and the results are compared with other encryption algorithms. They also discussed different masking techniques that protect from side channel attacks in boolean masking which is vulnerable to power analysis attack and some active attacks. So they implemented threshold masking techniques as a countermeasure to side channel and fault attacks (Bogdanov, Moradi, & Yalcin, 2012).

**Program Counter Security** This paper provides a requirement that when satisfied can guarantee safety against certain side-channel analysis attacks. It also provides tools that can test if code satisfies the requirement as well as tools that can transform code into new code that does satisfy the requirement (Molnar, Piotrowsi, Schultz, & Wagner, 2006).

**Cache-based SCA Defense**

This paper focuses on both access and time driven cache-based side channel attacks. The previous software only and hardware only approaches were effective in defending against either access or time driven but not both. The proposed three approaches in this paper defend both attacks and reduce the performance overhead (Kong, Aciiçmez, Seifert, & Zhou, 2009).

**4. Design/Architecture**

The software DES implementation uses the one available on the website Rosetta Code at this link: <https://rosettacode.org/wiki/Data_Encryption_Standard#Python>. Encryption and decryption use a 64 bit key and accept a 64 bit message as input. The key and message must both be a hexadecimal int object for the DES to execute. Standard DES procedure is followed otherwise From this base algorithm, our software implementation adds the functionality to use an ASCII string input of arbitrary length by converting the ASCII input into hexadecimal, segmenting the hexadecimal into 64 bit chunks, and padding any chunk the cannot reach 64 bits.

The hardware DES implementation uses HAPI DES, which is available on Github at this link: <https://github.com/IamVNIE/hapi-des>. A small Python script is used to transfer messages to a PYNQ-Z2 board and to return encrypted/decrypted messages from that board, but all DES operations are performed by hardware on the board. The implementation uses a 64 bit key and can accept an ASCII string of arbitrary length by use of a provided hardware data formatter. After being formatted, the ASCII string is divided into 64 bit chunks for encryption or decryption.

The 3DES encryption implementation works as follows. An arbitrary ASCII string input is first given to the hardware DES and is encrypted using the hardware DES key. The resulting output is then provided to the software DES and is encrypted using the first software DES key. The output of that is then provided once again to the software DES and is decrypted using the second software DES key, which results in the final encrypted string. The 3DES decryption implementation is simply the encryption implementation in reverse.

It is noted that our 3DES implementation does not follow the traditional 3DES format of encryption with key 1, decryption with key 2, and encryption with key 3. This is unfortunately due to time constraints. Given a little extra time, some simple modifications to our additional software DES functionality should allow the traditional format to work, but as the algorithm currently stands, the traditional format will fail. Our 3DES also has the restrictions that the input string must be ASCII, and that there could still potentially be strings that the hardware implementation could fail to encrypt. The strings that could cause this issue should be few and far between, but they are mentioned by the creator of the hardware implementation, so we felt the need to mention them as well.

**5. Evaluation/Experiments**

The following outputs were produced using a test run of our 3DES algorithm on the test string “This is a secret.” The input does not include the quotation marks, but it does include the spaces and the period, resulting in a 17 character long input. This means the string will have to be split into three pieces, and the last piece will need 7 characters of padding.

3DES Encryption:

1. Hardware DES Encryption took 0.00043201446533203125 seconds
2. Software DES Encryption took 0.029950857162475586 seconds
3. Software DES Decryption took 0.02558422088623047 seconds

3DES Decryption:

1. Software DES Encryption took 0.026334285736083984 seconds
2. Software DES Decryption took 0.026137351989746094 seconds
3. Hardware DES Decryption took 0.0004150867462158203 seconds

The explicit output of this test run can be found in our Github repository for this project as a part of the code file (DES\_program.ipynb - last block, at the bottom of the page.).

**6. Conclusions**

The creation of a hardware and software cryptographic algorithm was a success. Even though our implementation does not follow the traditional order of 3DES, the created algorithm still successfully encrypts a message using both hardware and software. However, 3DES is no longer considered a secure cryptographic algorithm, so this implementation is best used as a basis for creating a hardware and software implementation of a secure cryptographic algorithm like AES.

**References**

1. AlDosari, F. (2017). Security and Privacy Challenges in Cyber-Physical Systems. *Journal   
    of Information Security*, *08*(04), 285–295. <https://doi.org/10.4236/jis.2017.84019>
2. Bogdanov, A., Moradi, A., & Yalcin, T. (2012). Efficient and side-channel resistant   
    authenticated encryption of FPGA bitstreams. *2012 International Conference on   
    Reconfigurable Computing and FPGAs, ReConFig 2012*.   
    <https://doi.org/10.1109/ReConFig.2012.6416743>
3. Chevallier-Mames, B., Ciet, M., & Joye, M. (2004). Low-cost solutions for preventing   
    simple side-channel analysis: Side-channel atomicity. *IEEE Transactions on   
    Computers*, *53*(6), 760–768. <https://doi.org/10.1109/TC.2004.13>
4. Damopoulos, D. (2020, 1, 21). *Cybersecurity*. [PowerPoint Slides]. USAOnline.

https://ecampus.southalabama.edu/portal/site/b7472c65-cb38-4345-b900-06fe6402d288/tool/82d931e4-a5bd-4758-a5fc-97a06c82aa6e

1. IEEE Information Theory Society, & Institute of Electrical and Electronics Engineers.   
    (2012). *Information Theory Workshop (ITW), 2012 IEEE : Date: 3-7 September   
    2012 : [Lausanne, Switzerland]*. IEEE.
2. Khan, A. K., & Mahanta, H. J. (2014). Side channel attacks and their mitigation   
    techniques. 1st International Conference on Automation, Control, Energy and   
    Systems - 2014, ACES 2014, 1–4. <https://doi.org/10.1109/ACES.2014.6807983>
3. Kong, J., Aciiçmez, O., Seifert, J. P., & Zhou, H. (2009). Hardware-software integrated   
    approaches to defend against software cache-based side channel attacks.   
    *Proceedings - International Symposium on High-Performance Computer   
    Architecture*, 393–404. <https://doi.org/10.1109/HPCA.2009.4798277>
4. Lin, L., Kasper, M., Güneysu, T., Paar, C., & Burleson, W. (n.d.). *Trojan Side-Channels:   
    Lightweight Hardware Trojans through Side-Channel Engineering*.
5. Molnar, D., Piotrowski, M., Schultz, D., & Wagner, D. (2006). The program counter   
    security model: Automatic detection and removal of control-flow side channel   
    attacks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in   
    Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3935 LNCS*,   
    156–168. <https://doi.org/10.1007/11734727_14>
6. Picek, S., Heuser, A., Jovic, A., Ludwig, S. A., Guilley, S., Jakobovic, D., & Mentens, N.   
    (2017). Side-channel analysis and machine learning: A practical perspective.   
    *Proceedings of the International Joint Conference on Neural Networks*,   
    *2017*-*May*, 4095–4102. <https://doi.org/10.1109/IJCNN.2017.7966373>
7. Standaert, F.-X., Malkin, T. G., & Yung, M. (2009). *A Unified Framework for the   
    Analysis of Side-Channel Key Recovery Attacks*.   
    <https://doi.org/10.1007/978-3-642-01001-9_35> (Standaert, Malkin, & Yung, 2009)
8. Yang, Y., Wu, L., Yin, G., Li, L., & Zhao, H. (2017). A Survey on Security and Privacy   
    Issues in Internet-of-Things. *IEEE Internet of Things Journal*, *4*(5), 1250–1258.   
    <https://doi.org/10.1109/JIOT.2017.2694844>