The 19th Winona Computer Science Undergraduate Research Symposium

May 1, 2019

12:30pm to 2:30pm

Watkins 105

Winona State University
Winona, MN

Sponsored by the Department of Computer Science

at Winona State University

WINONA STATE UNIVERSITY

Computer Science Department

http://cs.winona.edu

Table of Contents

Title	Author	Page
IOT Security Analysis: Raspberry Pi Communication And Networking for Interfacing, Authentication, and Data Protection	**William Diedrick**	1
Natural Language Phrases in Lambda Calculus to be Converted to Generalized Constraint Language	**Matthew Dill**	4
Data Protection and Secure Encryption using RSA RSA and DES with IOT Devices	**Steven Granquist**	7
Integrated Server Documentation and User Monitoring	**Marshall Halleck**	12
Evaluation of JavaScript Photo Animation Applications	**Sean Joyce**	15
Using the Fourier Transform and Harmonics To Recreate Instrument Tonality	**Cameron Pelzer**	18

IOT Security Analysis: Raspberry Pi Communication and Networking for Interfacing, Authentication, and Data Protection

William Diedrick
Department of Computer Science,
Winona State University, 175 W
Mark St, Winona, MN 55987, USA
wdiedrick14@winona.edu

ABSTRACT

Abstract - Internet of Things (IoT) technology has changed the way data is collected and used – in this new wave of the digital industrial revolution. IoT technology currently has critical importance in manufacturing, service, transportation, and educational industries. Stakeholders focus on the infrastructure and information systems of IoT, leaving security to be less explored and developed. Lack of security leaves systems vulnerable to attacks from outside influence. Attackers invade company networks through vulnerabilities, costing millions of dollars in damages.

In this study, we aim to develop a secure Raspberry Pi communication system to help expand the understanding of effective and economic IoT solutions. Security tests oriented toward interface design, authentication, and data protection were used to find how secure a Raspberry Pi based IoT communication system is, and how to improve it. The purpose of this study is to improve the design of current IoT systems by designing Raspberry Pi IoT system oriented toward a secure connection for communication. The results of testing the system were compared to an effective device already available on the market - like Tangle. The design and use of a Raspberry Pi based IoT system was shown to be as effective in its security design as Tangle.

Categories and Subject Descriptors

[**C, Python**]: Language Contructs and Features – *Security, IOT, Analysis.*

General Terms

Performance, Design, Reliability, Experimentation, Security, Human Factors, Standardization, Theory.

Keywords

Security, Raspberry Pi, Encryption, IOT, Communication, Networking.

1. INTRODUCTION

Through this study we hope to achieve more information regarding the security capabilities of a Raspberry Pi. The Raspberry Pi is a small computer with both standalone and joined programming capabilities. It has evolved in capabilities such that it is becoming more relevant for businesses to purchase and use in their practices. One of the most frequent uses of such a device is to generate communication between other devices. This, along with any sort of device being used on the network, should be closely examined to identify any exploitable features that may exist. The Raspberry Pi is already being used by many companies to support their networks so networking should be a topic that is broadly documented. The high usage of such a device provides evidence that the developers of the device have been performing similar tests to ensure their product is safe for use by businesses. The Raspberry Pi is inexpensive when compared to other communication devices. The other communication tools are more specialized than the Raspberry Pi. Its design is purposely more flexible to allow for plenty of different software projects to be utilized within. It is this flexibility which may prove to be a weakness to its security down the line.

Comparable to the system design implemented on the Raspberry Pi is the IOT system of Tangle. The Tangle uses a string of individual transactions interlinked with each other and stored through a decentralized network of nodes to achieve secure communication in an IOT system. Tangle functions on the use of a blockchain-like use of Directed Acyclic Graphs (DAGs) to communicate between nodes. This process requires verification of two previous transactions within the network to insure secure communication for new additions to the IOT network. The Tangle is being used as a baseline to compare our Raspberry Pi security model to. The Tangle is insured to be secure within its interface, authentication methods, and data protection methods due to its practical and tested use within industrial systems today.

With this study we aim to identify any security risks that exist in the Raspberry Pi by testing it's networking capabilities and comparing it with Tangle's secure system design. This is accomplished by identifying the key security features revolving around interface, authentication, and data protection that exist in both systems and comparing them with Tangle's being a baseline of comparison. We will also provide an analysis of what kind of benefits and flaws the Raspberry Pi may have within a network system. This includes the basic methodologies of setting up a Raspberry Pi system to function fully and securely within an IOT system and what approaches are necessary to design a secure communication network.

2. HYPOTHESIS

The Raspberry Pi's and the network and communication assets are as secure as the Tangle's when it comes to its methodologies revolving around interface design, authentication, and data protection.

3. SECTIONS

In order to test the hypothesis, a design for a secure Raspberry Pi IOT system was created and tested by setting up three separate experiments designed to test the security of the device. These experiments targeted each of the critical security concepts of secure IOT devices; These were a secure interface, secure authentication, and effective data protection methodologies. Outlined below are the secure setup of the Raspberry Pi, the setup used in the experiment, what kind of tests were used to assess the security concepts outlined, and how these tests can be used within the experiment to interpret results to identify overall security of the Raspberry Pi system.

3.1 Secure Raspberry Pi Setup

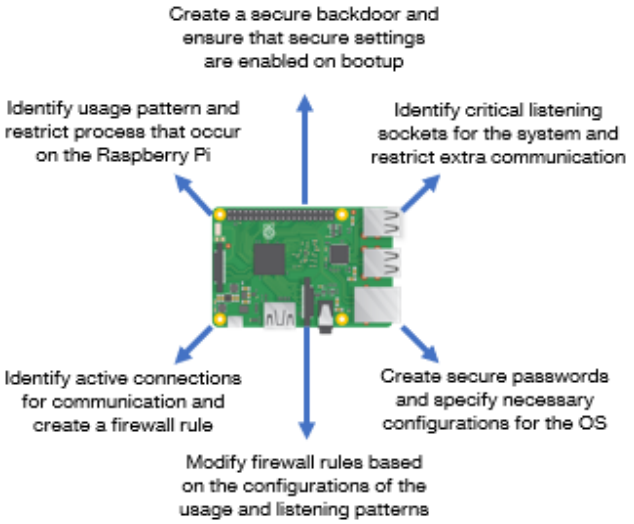


Figure 1: Secure Raspberry Pi Setup

The heading of subsections should be in Times New Roman 12-point bold with only the initial letters capitalized. (Note: For subsections and subsubsections, a word like *the* or *a* is not capitalized unless it is the first word of the header.) The Model of Raspberry Pi used in this test will be the Raspberry Pi 3, as it has the best configurations for Wi-Fi and communication methods. The Raspberry Pi runs Linux as a general-purpose operating system for full access to all of the services that exist within it. This brings risks due to the system being able to control much more than it needs to; Many of the features of the device must be limited to become more secure. The usage patterns of IOT systems are rigid and must be locked down [1]. The use of a script designed to identify what features are necessary for the IOT functionality, communication between two other devices in this case, will eliminate all unnecessary properties of the operating system. The necessary usage patterns of our system revolve around the process of forwarding information and the network connections involving listening sockets and active connections within the system. By setting up a system to frequently poll the processes and network sockets that exist within a single cycle of the system we can identify the most secure usage cycle of the system [2]. This usage pattern must be enforced through the use of a separate program designed to terminate any processes that exist outside of the patterns established. This is accomplished through a series of steps involving: Placing all the information we received from the earlier test into a readable file format, reading the file created and passing the information to program running on the Raspberry Pi, killing all processes not included within the usage pattern file within the IOT system, and creating a set of rules that are passed to the firewall to prevent unnecessary network interactions. Ensure your firewall only exposes the services you want, preferably on non-default ports [1]. These processes should be executed automatically upon startup of the system.

3.2 Experimental Design

The secure design for a Raspberry Pi setup within an IOT system is one part of a secure transaction experiment with other devices. This system requires not only the Raspberry Pi but also other devices in order to simulate the communication of an IOT system. Within this experiment the Raspberry Pi acts as a very simple information forwarding tool to avoid any outlying factors [2]. A network is formed using Bluetooth communication between two outside nodes and the Raspberry Pi, with the Raspberry Pi acting as a mediation tool between the nodes. Within the experiment this is simulated by using two separate computers with Bluetooth modules attached for communication. One computer acts as a "sender" which forwards information over Bluetooth to the Raspberry Pi which will receive it using its own Bluetooth module. The Raspberry Pi runs as securely as possible to then forward the information over Bluetooth to the second computer which acts as the "receiver" [3]. Both computers will be running simple communication programs to reduce risk of outlying factors interacting with the results of the experiment.

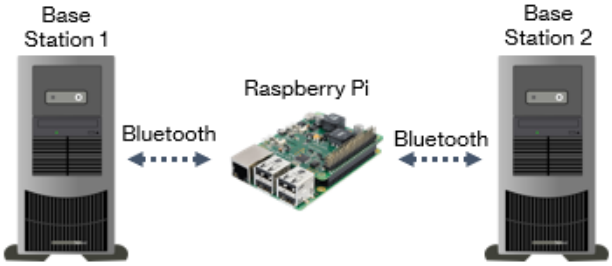


Figure 2: Experiment Setup

3.3 Testing

A set of three separate tests will be utilized to identify what kind of vulnerabilities exist within the Raspberry Pi system design and how well it compares to the security standards of the Tangle. Taking what we know of the Tangle and applying them to a Raspberry Pi system oriented toward communication we can gain more insight into how it compares to systems in use in modern industry. The first test targets the interface of the Raspberry Pi to insure that it is free of vulnerabilities that exist. We will create programs to run on the Raspberry Pi to initiate communication both through wired and wireless means. By utilizing wireless communication through Raspberry Pi Wi-Fi feature we will communicate with a base station. The Raspberry Pi will be running a version of Linux that is specified towards secure communication [4]. We will also develop programs designed to target the Raspberry Pi and see if there is any way to break security protocols. Direct attacks are used to test the security features of IoT Chain, we will use these for the Raspberry Pi as well. We can see how successful this experiment is based on our ability to exploit the security features of the Raspberry Pi. If we successfully defend from device attacks, we can say that the Raspberry Pi is secure enough to be used in communication systems. If there any exploitable security issues, we can say that it

is not secure enough. The second test targets the safe communication and authentication protocols of the device. Using symmetric encryption AES algorithms we will encrypt test messages which will be transferred between the chain of devices. Insuring that the message can be understood by the Raspberry Pi while not being exposed to outside influences is the key to this process. The final test is targeting data protection. For this test we will ensure that there are no problems that exist within protecting the data that is being transferred within the IoT system. Essentially this will be tested by ensuring that the message to be transferred is not stored anywhere on the Raspberry Pi. The information should be removed from the system in a manner that does not expose it to any outside attacks.

3.4 Interpretation

The tests that have been set up for the system are designed to compare standards already set by the Tangle IoT system. By using what makes this secure system successful in a market-oriented environment, we can compare the Raspberry Pi to see if it can also be used. If the Raspberry Pi can successfully pass the security tests that are set up within the system, the secure aspects of the device will become apparent.

4. CONCLUSION

The tests that have been set up for the system are designed to compare standards already set by the Tangle IoT system.

- **Interface:** The Raspberry Pi interface can be successfully configured to prevent unnecessary communication. Sockets and active connections are configured to a state that is similar to Tangle. Firewall configurations provide security from outside threats.
- **Authentication:** The Raspberry Pi successfully encrypts, and decrypts information being passed through the system. This process functions comparably to the standard set by Tangle.
- **Data Protection:** The Raspberry Pi does not store information from the communication in significant locations. No access is allowed to storage because of the secure setup similar to Tangle.

Through the security tests, the Raspberry Pi IOT design has been shown to be as effective in its security design as a device already available on the market, Tangle. Further testing for additional security features is necessary to ensure that a design like this could be used within a marketable system.

5. FUTURE WORK

There are some objects that should be addressed in the future regarding this research. This test was oriented specifically toward the Raspberry Pi device, in the future these tests should be adapted so that it fits a more broad spectrum of devices. The tests should also be applied to more large scale projects. The testing environment used for this project was small and could be expanded for future works. Only certain security principles were targeted for this research. If the research were to be concluded, all aspects of security should be defined and tested on the Raspberry Pi Setup addressed in this paper.

6. ACKOWLEDGEMENTS

William Diedrick thanks the WSU Computer Science Department, Dr. Gerald Cichanowski, and Dr. Sudharsan Iyengar for their help with this project.

7. REFERENCES

[1] Keane, Justin Klein. “Take These Steps to Secure Your Raspberry Pi Against Attackers | Make:” Make: DIY Projects and Ideas for Makers, Make: Projects, 7 Sept. 2017, makezine.com/2017/09/07/secure-your-raspberry-pi-against-attackers/.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Mercer, Dan. “Securing Your Raspberry Pi.” RaspberryPi, 28 Dec. 2018, www.raspberrypi.org/documentation/configuration/security.md

[3] “Teach, Learn, and Make with Raspberry Pi.” RaspberryPi, www.raspberrypi.org/.

[4] Yutanto, Hariadi. “SECURITY INTELLIGENCE FOR INDUSTRY 4.0: DESIGN AND IMPLEMENTATION” International Scientific Journal 65.09 (2018). Academia. Web. 21 January 2019.M.

Natural Language Phrases in Lambda Calculus to be Converted to Generalized Constraint Language

Matthew Dill

Computer Science, Winona State University
175 W Mark St, Winona, MN 55987, United States
mdill15@winona.edu

Abstract

This study explores one aspect of bridging Computing with Words with Natural Language Processing, to connect the extraction capabilities of Natural Language Processing with the inference capabilities of Computing with Words. A program was written to convert a logic-based lambda calculus representation of an English natural language expression into General Constraint Language. The scope of this project is set to only to simplistic expressions, and is a foundation for expanding upon more complex lambda calculus expressions into General Constraint Language. This program tags the parts of speech from the lambda calculus expression and outputs the General Constraint Language of the expression, showing the constraint on an idea in the original sentence. The program is limited in functionality, and can only tag parts of speech in simple lambda calculus expressions. This project establishes an entry point, and is designed with further improvements and modifications in mind.

1. Introduction

1.1 Background

Natural language processing and computing with words are rapidly advancing fields, with many advancements in A.I. and ways to work with natural languages on machines being made each year. Natural language processing has been a major focus in artificial intelligence for many years, and computing with words is a more recent field. Being introduced in 1965 by Dr. Lotfi Zadeh, with a paper he published on fuzzy logic and the machinery to process words [1]. Another aspect of this is using lambda calculus to store natural language expressions, and thus be able to process them on a machine as a lambda calculus expression [2].

Proceedings of the 19th Winona Computer Science Undergraduate Research Seminar, May 1, 2019, Winona, MN, US.

1.2 Goal of Study

This study would be bridging these two areas, and would utilize several components of the field, such as parse trees, fuzzy logic, and deductive reasoning. The primary goal is to write a translator that takes a lambda calculus expression representing a natural language expression, and converts it into Generalized Constraint Language (GCL). This program will use elements of natural language processing and decision theory, such as interpreting sentence structure and degrees of truth, to accurately translate the expression into GCL [3].

1.3 Generalized Constraint Overview

On a fundamental level, GCL is formatted as X *is* R, where X is a linguistic variable or another Generalized Constraint expression, and R is the constraint on X [4]. As mentioned earlier, GCL is primarily used for its inference qualities, so the purpose of forming these expressions is to provide meaning to the keywords defined in the lambda calculus. By easily being able to calculate the GCL expression from the lambda calculus, the theoretical program that is using this will be able to save resources computing the GCL, and will be able to have the keywords, relationship(s), and meaning of the words faster than computing both the lambda calculus and GCL separately [5].

Another part of this project is specifying the modality of the sentence in the GCL expression. The modality of a natural language expression is, put simply, the semantic meaning of the expression. This provides an additional layer of meaning to the sentence, and can help programs processing natural language input by providing that extra meaning. Functionally, this is represented by a fuzzy number inputted alongside the natural language phrase. In this study, the modality and the appropriate fuzzy number will be tagged in the lambda calculus expression [5].

1.4 Contribution to the Field

This program will be significant to the field because it will allow for a current representation of natural languages to be easily converted to another form that can be applied to CW. This is important because GCL is helpful in CW for making expressions easy to process and interpret. It will be a meaningful contribution to set a basis for future developments, and to further the advancements in the field.

2. Hypothesis

A program can be written to convert a natural language expression stored as lambda calculus into Generalized Constraint Language.

3. Method

This project uses Java for the program, and uses Strings for the input. The program is designed around using natural language processing concepts to tag the parts of speech from the lambda calculus expression, and then output the meaning of the relationship between the subject and the predicate of the provided sentnece. The development of the program follows the agile model, with improvements and additions being made throughout the implementation process. Testing the program is simplistic, as it is running the program on various lambda calculus expressions to make sure the correct parts of speech are tagged and outputted.

4. Results

Table 1: Example Lambda Calculus Inputs Tested

Sentence	Lambda Calculus	Generalized Constraint Language
The robot is close to the wall.	(λx.λy.distance(y, x))	Distance of robot *is* close
The tank of gas is full.	(λx.λy.status(y, x))(v.0.7)	(Status of tank of gas *is* full) *isv* somewhat true
The countertop is clean.	(λx.λy.condition(y, x))(v.0.2)	(Condition of countertop *is* clean) *isv* hardly true
The sun is bright.	(λx.λy.status(y, x))(v.1.0)	(Status of sun *is* bright) *isv* certain
The car is red.	(λx.λy.color(y, x))(u.0.65)	(Color of car *is* red) *isu* usually
The food tastes good.	(λx.λy.taste(y, x))(u.0.45)	(Taste of food *is* good) *isu* not usually
The groceries contain milk.	(λx.λy.contents(y, x))(p.0.8)	(Contents of groceries *is* milk) *isp* likely
The person walks fast.	(λx.λy.speed(y, x))(p.0.5)	(Speed of person *is* fast) *isp* unlikely
The cat is blue.	(λx.λy.color(y, x))(p.0.0)	(Color of cat *is* blue) *isp* impossible

Table 2: Explanation of Modalities Used

Modality	Meaning	Example GCL
Blank	Shows direct relationship	Distance of robot *is* close
Verity (v)	Shows truth of the statement	(Status of tank of gas *is* full) *isv* somewhat true
Usuality (u)	Shows probability through basic likelihood of the statement	(Taste of food *is* good) *isu* not usually
Probability (p)	Shows probability of the statement	(Contents of groceries *is* milk) *isp* likely

5. Analysis

5.1 Analysis of Results

This study focuses on simple natural language expressions, and as such simple English sentences are the basis for the lambda calculus input. All sentences used consisted of a one subject and one predicate. In Table 1 and Table 2 (shown above), example sentences are provided. Using this simple sentence structure, any similar sentence can be used as the basis for the lambda calculus input of the program.

The GCL output of the program is as expected, and shows the correct modality of the inputted lambda calculus expression. Since the input is based on simple sentences, the GCL output is not overly complex, but it does highlight the basic meaning and relationship between the keywords of the given sentence. In future research in this area, this conversion can be expanded to work with more complex sentences, which will greatly increase the capabilities of the program.

5.2 Analysis of the Program

In this program, given that the sentences are simple, the parts of speech are easily identifiable. Since the input had the subject assigned to the x variable in the lambda calculus and the predicate assigned to the y variable, no natural language processing is needed to identify the parts of speech. For more complex sentences and input, this would be necessary, and is something that is a plan for future work on this project.

When the program runs and takes in the given lambda calculus expression, the program formats the relationship between the subject and the predicate. In addition to this, the program also highlights the modality of the relationship, which is specified in the fuzzy number added at the end of the lambda calculus. By combining these two elements, the program is able to produce a GCL expression that provides accurate inference to the machine about the meaning of the phrase the inputted lambda calculus is based on. The modalities explored in this study are not all of the modalities defined by Dr. Zadeh. Since this project only used simple lambda calculus expressions, some of the more complex modalities were not included in the program's functionality. The three used, those being verity, usuality, and probability, all provided a significant additional

meaning to the provided sentences, and were reasonable for the simple sentences used in this project.

6. Conclusion

In conclusion, the program written in this study was able to successfully convert simple lambda calculus expressions into Generalized Constraint Language. The GCL outputted accurately showed the relationship between the subject and the predicate of the given sentence, and correctly identified the modality present in the sentence. While the inputs used were simple, establishing this link between lambda calculus and Generalized Constraint Language is useful for allowing greater interpretive qualities in machines and programs that process natural language statements. Going forward, more features and capabilities will be explored, such as including the functionality for more of the currently defined modalities, as well as allowing for more complex sentence input.

7. References

[1] Zadeh, Lotfi A.. "Fuzzy logic = computing with words." *IEEE Trans. Fuzzy Systems* 4 (1996): 103-111.

[2] Barendregt, Henk. "The Impact of the Lambda Calculus in Logic and Computer Science." *The Bulletin of Symbolic Logic*, vol. 3, no. 2, 1997, pp. 181–215. *JSTOR*, www.jstor.org/stable/421013.

[3] Zadeh, Lotfi. *Computing with Words Principal Concepts and Ideas*. Springer Berlin, 2014.

[4] Khorasani, Elham & Rahimi, Shahram & Calvert, Wesley. "Formalization of Generalized Constraint Language: A Crucial Prelude to Computing With Words." *IEEE transactions on systems, man, and cybernetics*. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society. 2012. 43. 10.1109/TSMCB.2012.2204743.

[5] Zadeh, Lotfi A. "Toward a generalized theory of uncertainty (GTU)—an outline." *Information Sciences*, vol. 172, no. 1–2, 2005, pp. 1-40, http://www.sciencedirect.com/science/article/pii/S00200255050 0054X.

Data Protection and Secure Encryption using RSA and DES with IOT Devices

Steven Granquist
Adviser: Shimin Li
Computer Science, Winona State University, 175 W Mark St, Winona, MN 55987, United States
1- 651-900-3654
[1]sgranquist15@winona.edu
[2]shiminli@winona.edu

***Abstract*— This paper examines different methods of encryption and Data Protection on Internet of Things (IoT) devices. As IoT devices become more popular, they need to be properly secured. Specific comparisons are made between RSA and DES encryption and how those methods handle the processing of information and data across different types of devices. The goal of the paper is to determine which method is the most efficient and computationally effective to be used with IoT devices. RSA and DES were imitated using a Raspberry Pi, a Laptop, and a Desktop PC. OpenSSL was used to encrypt those devices with RSA and DES. Based off the experiments done, RSA encryption is the method that would be best suited for implementation on IoT devices from a computational perspective.**

***Keywords*— IoT, Encryption, Data Protection, RSA, DES, Security.**

I. Introduction

Internet of Things (IoT) devices are becoming more prevalent as we step into the future of technology. These devices are becoming ingrained in culture and the daily life of millions of people. Where there is a large amount of people, there is bound to be a security issue. IoT devices post a large amount of risks in their current state, many devices are un-patchable and incredibly vulnerable to different types of intrusions.

According to predictions from Statista, the total number of IoT devices will reach around 31 billion devices in 2020. By 2025, this number is estimated to reach around 75 billion [1]. This massive variety of interconnected devices serves to be a potential threat if not handled properly. “Everything from the physical or virtual world will possible be connected by the IoT. Connectivity between the things shall be available to all with low cost and may not be owned by private entities. For IoT, intelligent learning, fast deployment, best information understanding and interpreting, against fraud and malicious attack, and privacy protection are essential requirements” [2].

Properly encrypting these devices is the next step we must take to ensure we set ourselves up for success in the future. In a study done with the most popular IoT devices, and average of 25 vulnerabilities were found [3]. As we use more and more IoT devices over time, we need to make sure we are taking the proper precautions to protect and properly secure these devices and their users, our devices should have 0 vulnerabilities for them to be deemed secure and ready for use.

There are many ways these devices can be secured, and many types of encryption currently exist that could possibly be used to solve this problem. One problem that arises when trying to address this issue is the relationship that all these devices have with and to each other. Many IoT devices use a variety of software, operating systems, and protocols to perform their functions, this makes it harder to develop a universal solution, therefore, harder to enable a standard set of encryptions. There are many different encryption styles that can be considered as a possible solution for this issue, however, this study will focus specifically on two different methods and how they compare to each other. Based on these comparisons, we can determine which method would be ideal to implement in environments similar or identical to IoT devices. The two methods of interest being explored in this paper are RSA and DES. While both methods are very different, comparisons can still be made between them. Utilizing OpenSSL, RSA and DES encryption was implemented and extensively tested. For RSA, different key lengths are

Proceedings of the 18thWinona Computer Science Undergraduate Research Seminar, May 1, 2019, Winona, MN, US.

compared. For DES, 4 different methods of block ciphers will be compared. When comparing the key lengths, results can be a little bit misleading, [4] an accurate comparison between two types of encryption may need to use two completely different key lengths, in order to be fair to the different lengths of RSA. Lower levels of RSA, such as 1024 and 2048 length methods, can be compared to the DES methods shown later in the paper because of the similar level of security. Although RSA and DES are two different types of encryption, they still can be accurately compared to each other.

II. HYPOTHESIS

How do the different methods perform on different devices of varying hardware and software configurations? Do some OS's handle the encryption better than others?

Between RSA and DES, which method could be the best solution to properly secure and protect IoT devices and their critical data? And how can we verify this method is better than the other?

III. METHODS

The two methods of encryption chosen, RSA and DES, have been implemented to run on multiple devices with a variety of hardware and software constraints. These devices consist of the following; a Raspberry Pi running Raspbian OS, a HP Laptop running Debian on Windows 10, and a Desktop PC running Ubuntu. With these devices, the two methods can be tested with a variety of constraints and options to narrow down which encryption style offers the best performance and overall protection.

The devices vary in operating system, which was considered. While a bias towards a certain OS was likely to exist and be found [5], no observable evidence was found to prove this point.

A. Experiment on Device

For an individual experiment on a device, the process was as follows: In a project file, five different sizes of files were created. These files each had a different amount of characters within them, this allows an average to be computed for the overall effectiveness of the method. The five different ranges of character selected were, 6,000, 18,000, 42,000, 63,000 and 280,000. Using OpenSSL, the encryption method was specified, either RSA or DES. The above five files were encrypted, then decrypted. Time to complete both tasks were measured, in addition to the total overall time to complete. The encryption and decryption process were done five times for each of the five files. Given the results from the five trials, an average was computed, this average was said to be the average encrypt time and the average decrypt time. A total of 25 trials for each file, were summed and averaged to determine the average time for a given method to run across a variety of files.

The above steps were followed using RSA (1024), RSA (2048), RSA (3072) and RSA (4096) for RSA. These key values selected are in bits. For DES, DES, DES-CBC, DES-ECB, and DES-EDE block ciphers were tested. These RSA and DES methods were tested on all three of the test devices. The results can be viewed below in the results section.

B. DES

Pertaining to DES, four different block cipher methods were utilized. A brief explanation is needed to describe the differences between the four DES methods. DES uses two fundamental attributes of cryptography: Substitution (confusion) and transposition (Diffusion). [6]. DES is a block cipher; it encrypts the data in a block of 64 bits. The key length is 56 bits, after certain bits are discarded as they are unneeded [6].

C. DES-CBC

DES-CBC uses DES as the base of the algorithm, with Cipher Block Chaining, Like DES, DES-CBC uses a 64-bit key. 56 of those 64 bits are used directly, the other 8 are odd parity bits [7]. DES-CBC also required a 64-bit Initialization Vector (IV). For each encrypted message or data, a new IV is generated. This IV is sent with the message within the header field, included with the 64-bit key [7].

D. DES-ECB

DES-ECB uses DES as the base of the algorithm, with what is described as an Electronic Codebook, hence the ECB. In this mode, the data being encrypted is divided into different, individual variable-sized blocks. These separate blocks are then encrypted on their own. The blocks do not need each other in order to decrypt the encrypted message, a specific block just needs its own individual key to decrypt its information [8]. All the blocks hold the entire original message, but they each hold their piece of the message. Later, these blocks will be recombined to remake the original message. While this method does extra encryption within the message, it is not very good at it. Most instances of DES-ECB do not hide data patterns very well. Below is an example of how clear the message can be viewed despite being encrypted.

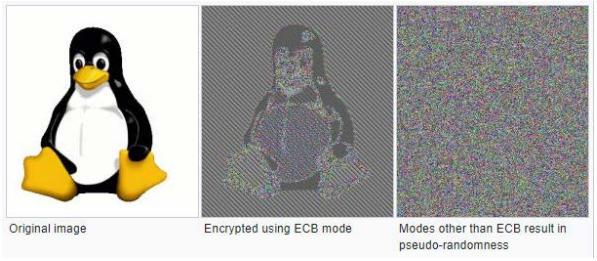


Fig. 1 DES-ECB example [8].

E. DES-EDE

DES-EDE uses an Encrypt-Decrypt-Encrypt, EDE, multiple encryption mode [9]. It uses multiple pairs of 64-bit keys, instead of just one like normal DES. The pair of keys is later passed in the header field to be used later [9].

F. RSA

Using the RSA key lengths was easy to implement, unlike what was originally predicted. As expected, with the RSA key length increase, computation time also increased. RSA 2048 was previously considered to be the standard for RSA encryption, that is now moving more towards the large key lengths, 3072 and 4096. The NSA has determined the RSA with a minimum key length of 3072 is deemed safe enough to protect information up to Top Secret information [10]. This decision was made based off the NIST's document, SP 800-56B Rev. 1, published to demonstrate the level of security that RSA with 3027 and 4096 provides [11].

RSA algorithm is an asymmetric based cryptography method [12]. This means that there are two different keys, the Public Key and the Private Key. An example of this relationship is as follows [12]:

1. A client (a web browser) sends its public key to the server and requests for some information.

2. The server encrypts the data using the client's public key and send the encrypted data.

3. The client receives this data and decrypts it.

Even though the client gives up its key, it is not helpful to anyone listening in hopes of stealing information, since the public key is not used to decrypt the information. This means that information can be safely sent without worrying if someone in the middle is intercepting the encrypted message [13]. However, RSA is not perfect, "methods such as brute-force are simple but lengthy and may crack a message, but not likely an entire encryption scheme. It cannot be verified that RSA is unbreakable… Despite years of attempts, no one has been known to crack the RSA algorithm" [14]. The reason this algorithm is so effective is because of how it is made. "RSA cryptosystem is based on the dramatic difference between the case of finding large primes and the difficulty of factoring the product of two large prime numbers" [15].

In the experiment, a Public and Private key were generated in order to encrypt and decrypt the original data file. The client and server were both on the same device, no actual transmission was made with the data files, but the files were encrypted with the Public/Private key pair.

Table 1. RSA Encryption

Device	**Type (in bits)**	**AVG Total Time (Seconds)**	**AVG Total Time (Milliseconds)**
Pi	1024	0.460	460
Pi	2048	0.471	471
Pi	3072	0.482	482
Pi	4096	0.489	489
Laptop	1024	0.165	165
Laptop	2048	0.168	168
Laptop	3072	0.175	175
Laptop	4096	0.184	184
Desktop	1024	0.732	73.2
Desktop	2048	0.756	75.6
Desktop	3072	0.796	79.6
Desktop	4096	0.843	84.3

Table 2. DES Encryption

Device	**Type (in bits)**	**AVG Total Time (Seconds)**	**AVG Total Time (Milliseconds)**
Pi	DES	0.9024	902.4
Pi	DES-CBC	0.9055	905.5
Pi	DES-ECB	0.9329	932.9
Pi	DES-EDE	1.0568	1056.8
Laptop	DES	0.316	316
Laptop	DES-CBC	0.3188	318.8
Laptop	DES-ECB	0.3246	324.6
Laptop	DES-EDE	0.3511	341.1
Desktop	DES	0.0916	91.6
Desktop	DES-CBC	0.0939	93.9
Desktop	DES-ECB	0.0946	94.46
Desktop	DES-EDE	0.1032	103.2

IV. Analysis

DES: Based on the result from the experiment, basic DES has been determined to be the standout choice between the DES methods as the most efficient method. Across all the test cases and devices, DES always had the lowest Average Total Time in both seconds and milliseconds. When DES times are compared to the other DES block cipher methods, DES beats them all in terms of runtime. In most test cases, DES-CBC came in a very close second, and seemed like a possible choice, however, after extra testing between DES and DES-CBC, DES proved to have the fastest average time. DES-EDE was the method that took the longest time on all three of the devices. This is caused by the fact that DES-EDE is a multiple encryption mode, with multiple 64-bit keys, which causes the average total time to be larger. While all the DES methods ran in under 1 second, DES is still the best choice out of the 4. Below is a graph that shows this relationship:

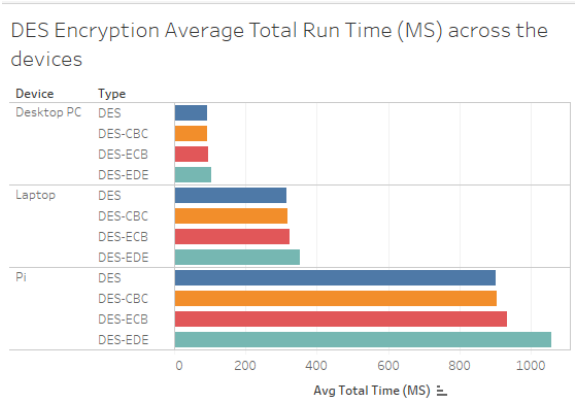


Fig. 2 DES Encryption Average Total Run Time

RSA: Based on the result from the experiment, all methods of RSA can be used as a viable choice as the most efficient method. Across all the test cases and devices, all key lengths in RSA had relatively low Average Total Times. In all cases, the results had a very similar relationship. Regardless of key length, the times for RSA were the same on each device. For the Desktop PC, all the tests resulted in a time (in MS) between 70-80. For the Laptop, all the tests resulted in a time (in MS) between 160-180. For the PI, all the tests resulted in a time (in MS) between 460-490. Technology with more processing power can run RSA more efficiently and obtain more smaller times. As the level of processing power goes down, there is an increase the Average Total Time range. Below is a graph that shows this relationship:

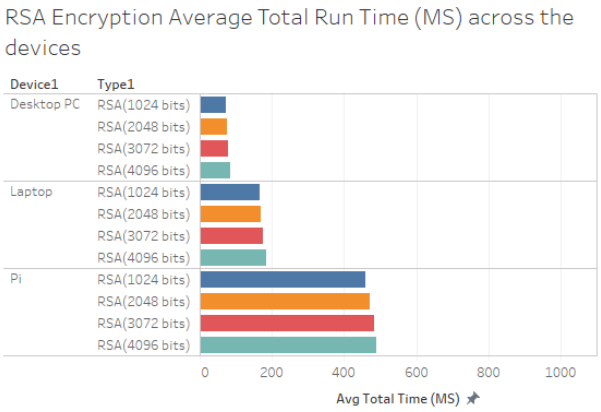


Fig. 3 RSA Encryption Average Total Run Time

RSA vs DES: When comparing RSA vs DES, it becomes clear that RSA is faster at all levels when compared to DES. Even though DES encryption is simpler than RSA, RSA still runs faster in every test case. In some cases, like the Desktop PC, the differences are very slim. In the Laptop and Pi tests, the differences are very drastic. For the Laptop and Pi, DES takes twice as long compared to RSA. It becomes clear when comparing the Average Total Times between RSA and DES, that RSA is the better choice in every test case. Based off this information, RSA encryption is the method that would be best suited for implementation on IoT devices from a computational perspective. Below is a graph that shows this relationship:

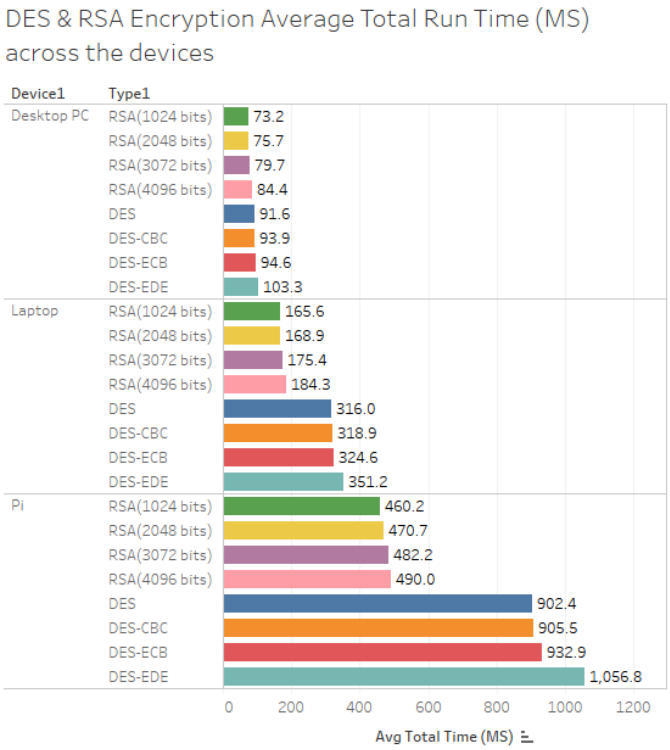


Fig. 4 RSA vs DES Encryption Average Total Run Time

V. Conclusion

Based on the results from the experiments, the best method to implement on IoT devices was able to be reliably determined. RSA encryption method is the standout choice based on the results from the experiments. RSA was the most efficient at runtime on each of the test devices. In every case, RSA performed faster than DES in terms of completion time. This means that in smaller IoT devices, RSA encryption is viable at initial setup of the device. This method is the most effective across the various hardware and would be good to use in environments with large transportation of data. There was not a bias between the OS's of the devices. This can be verified by the fact that all the test results followed a similar pattern. All DES methods were close between each other; however, RSA was significantly faster than DES. Based on all these parameters, it can be reliably determined that RSA is the best solution to properly secure and protect IoT devices and their critical data.

References

[1] "Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions)". Statista, Nov-2016. [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

[2] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A Vision of IoT: Applications, Challenges, and Opportunities with China Perspective," IEEE Xplore, 09-Jul-2014. [Online]. Available: https://ieeexplore.ieee.org/document/6851114.

[3] E. Bertino, "Data Security and Privacy in the IoT," OpenProceedings.org, 15-Mar-2016. [Online]. Available: https://openproceedings.org/2016/conf/edbt/paper-a.pdf.

[4] M. Suarez-Albela, P. Fraga-Lamas, and T. M. Fernandez-Carames, "A Practical Evaluation on RSA and ECC-Based Cipher Suites for IoT High-Security Energy-Efficient Fog and Mist Computing Devices," ResearchGate, Nov-2018. [Online]. Available: https://www.researchgate.net/publication/328890429_A_Practical_Evaluation_on_RSA_and_ECC-Based_Cipher_Suites_for_IoT_High-Security_Energy-Efficient_Fog_and_Mist_Computing_Devices.

[5] K. Deshpande, P. Singh, "Performance Evaluation of Cryptographic Ciphers on IoT Devices," arxiv.org, 5-Dec-2018. [Online]. Available: https://arxiv.org/pdf/1812.02220.pdf.

[6] Nirm Aljeet Kaur, S. Sodhi, "Data Encryption Standard Algorithm (DES) for Secure Data Transmission," research.ijcaonline.org. [Online]. Available: https://research.ijcaonline.org/icaet2016/number2/icaet036.pdf

[7] "DES in CBC Mode (DES-CBC)", freesoft.org. [Online]. Available: https://www.freesoft.org/CIE/RFC/1423/2.htm

[8] "Block cipher mode of operation", wikipedia.org. [Online]. Available: https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#Electronic_Codebook_(ECB)

[9] "DES in EDE Mode (DES-EDE)", freesoft.org. [Online]. Available: https://www.freesoft.org/CIE/RFC/1423/8.htm

[10] National Cryptography Solutions Management Office, "Commercial National Security Algorithm Suite", nsa.gov, 19-Aug-2015. [Online]. Available: https://apps.nsa.gov/iaarchive/programs/iad-initiatives/cnsa-suite.cfm

[11] Computer Security Division (Information Technology Laboratory), "Recommendation for Pair-Wise Key-Establishment Schemes Using Integer Factorization Cryptography", nist.gov, Mar-2019. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br1.pdf

[12] "RSA Algorithm in Cryptography", geeksforgeeks.org. [Online]. Available: https://www.geeksforgeeks.org/rsa-algorithm-cryptography/

[13] R.L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", csail.mit.edu. [Online]. Available: https://people.csail.mit.edu/rivest/Rsapaper.pdf

[14] Evgeny Milanov, "The RSA Algorithm", math.washington.edu, 3-Jun-2009. [Online]. Available: https://sites.math.washington.edu/~morrow/336_09/papers/Yevgeny.pdf

[15] M. Preetha, M. Nithya, "A STUDY AND PERFORMANCE ANALYSIS OF RSA ALGORITHM", ijcsmc.com, 6-Jun-2013. [Online]. Available: https://www.ijcsmc.com/docs/papers/June2013/V2I6201330.pdf

Integrated Server Documentation and User Monitoring

Marshall Halleck
Winona State Computer Science
175 West Mark Street
Winona, Minnesota 55987
1-(507)-457-5375
Mhalleck10@gmail.com

ABSTRACT

Lack of detailed server documentation affects performance of employees managing servers and services they provide. Proper and relevant documentation can help reduce resolution time. Extensive support for documentation in software development is common, however there remains a lack of such for servers. Accumulated and properly collated documentation from services provided can be a great repository for future use and can help streamline services.

To support the aforementioned accumulation, we implemented a required justification of services/work done on exit time. This can be achieved via a number of open-source tools, including sysdig for user monitoring, mySQL for database management, and Java Database Connectivity for simplifying Java interfacing with the database. Implementation will occur on a series of Debian-flavored Linux virtual machines. Success was measured via user surveys and documentation rate. Surveys reflect a positive attitude, with documentation rate climbing over time. Users found the generated documentation helpful for future services, however utilization remains low due to poor database querying capabilities. For the future we aim to fully implement a web-interface for querying support and begin measuring direct impact of documentation on service performance time.

Keywords

documentation, tool, user monitoring, computer science, d ebian, linux, sysdig, sql, java, eclipse

1. INTRODUCTION

TAs technology grows and evolves, the sheer quantity of services offloaded to servers has equally increased. Servers are not only becoming more complex, but the number of and location makes it almost unmanageable to directly interface with servers; combined with now affordable cloud-hosting models like Amazon Web Services, direct interfacing becomes impossible. This has led rise to monitoring tools: distributed services running on servers reporting on health. These tools allow for both hardware and software monitoring, usually combined with log scanning and monitoring as well. Allowing for vastly simplified management, monitoring tools have become not a quality of life, but a necessity for many production environments. Systems grow, and inevitably so do the number of people required to manage them. Needed to cope, growing server management departments exacerbate a common theme in Computer Science, lack of documentation. Well-written documentation trivializes team project communication, with all requisite information being readily available. This remains true for any manner of project, from software development to server management. The majority of documentation tools focus on software development with none supporting other classes of Computer Science expertise to the same extent. For server management this rings true. Changelogs currently provide the only semblance of documentation tools, with most management groups relying on, in the best case, shared databases, and the worst case, none at all. Monitoring tools have evolved to deal with growing server service requirements, however no documentation still remains an independent task, interrupting work flow and deincentiving users from creating new documentation.

Server managers perform the majority of server interfacing via a Secure Shell, or a remote way of executing commands on a server. For the purposes of this paper these managers are the 'users' of the system. Users need only interact with these servers for either training purposes, or for providing services, like updating firmware, configuring software, or troubleshooting errors. While many simple service scenarios are well documented, the uniqueness of hardware and software configurations in actual production environments requires user generated document for posterity's sake. A well documented history of services and their impact provides a record for troubleshooting, ensuring services were properly performed and executed on time, and provide an extensible basis for more detailed documentation in the future. Lack of documentation can be attributed to a number of different factors including lack of training and reinforcement, however as we learned from software development, by integrating documentation with the development process both the quality and quantity of documentation increased. [1] [2] [3]

2. HYPOTHESIS

Integrated documentation and user monitoring will streamline the documentation process and optimize services.

3. Development Methodology

The software was developed using the agile method to account for changing project definitions, goals, and to accommodate future updates. Java development occurred in the Eclipse IDE, developed for and implemented in the Debian 9 Linux environment.

3.1 Software Components

Our software consists of three main components; first, a way to log user events and request documentation. Second, a way to send this information from all the monitored servers to a database. And third,

Proceedings of the 19th Winona Computer Science Undergraduate Research Seminar, May 1, 2019, Winona, MN, US.

a way to query the database, allowing users to see their generated documentation.

3.2 Logging Service

The logging service tracks user login and logout times, user commands, then ultimately requests documentation. The logging service uses the login. As each event occurs it is piped via a file to the client service.

3.2.1 Login-Logout

Login and Logout are trivially tracked by the built-in scripting capabilities of Linux. The login script also begins tracking user commands, with the logout script terminating the tracking.

3.2.2 User Commands

The user login script activates the open source utility *sysdig*[1] to begin tracking user commands.

3.2.3 Documentation Request

To properly request documentation on logout, we defined a custom exit script 'doc' for users to run at service completion. This script performs two tasks, first, requesting the user to briefly type a description of their provided service. On hitting 'enter' the script pipes the documentation to the client and disconnects the user.

3.3 Client Server Messaging

To support more than a single server, client-server messaging is required, requiring the server to procedurally add/remove clients as the come online.

3.3.1 Client

The client receives via the file pipe, all user events. As these are intercepted, they are associated with server specific metadata and a service ID, encrypted, and sent to the server. Many client instances may exist at a time, with each manually configured with the proper host IP.

3.3.2 Server

Only a single instance of the server exists at once. The server process listens on a desired network port for incoming client connections. The server accepts two types of traffic, authenticated via encryption, or initial client connections. On client connect, a key is generated by the encryption service and the server shares this with the client.

3.3.3 Encryption

Encryption is performed via built in cryptographic Java key generation libraries, currently implementing symmetric AES. The encryption process generates keys as requested, and encrypts messages with an assignable key. Key's are encrypted and stored in the database while not in active use.

3.4 Database and Interface

A relational database makes the most sense for our purposes, supporting advanced querying methods if necessary, and allowing for trivial association of documentation with meta-data.

3.4.1 Database

A simple mySQL relational database was used, with the Java Database Connectivity library to allow the server process to directly interface.

3.4.2 Interface

Due to limited development time, only primitive database interface techniques are implemented, with only precise SQL queries supported.

Table 1. Survey Results (Simplified)

4. TESTING METHODOLOGY

Our software was tested on the eight Linux servers currently in production at Winona State, managed asynchronously by three student employees. We aimed to measure both quantitative impact via documented server accesses, and qualitative via user surveys. Documented server accesses are measured as the rate of documented to undocumented accesses, with 0% reflecting no documentation, and 100% reflecting total documentation coverage. User surveys queried general satisfaction, usability, and perceived impact on service performance, all on a 1-10 scale, with additional space for general feedback. Surveys were given two weeks after initial implementation.

5. RESULTS

General feedback shows a high degree of impact on provided services, with a number of new, now deemed necessary, foundational documents generated as a result. Criticisms surrounded the querying tool, with users finding the SQL querying capabilities limiting, both due to the mathematical and precise nature of the queries and being required to access the server hosting the database to make said queries. At two weeks, documented user accesses rests at around 65% per day.

UserID	General Satisfaction	Usability	Perceived Impact on Services
SG	7	4	8.5
TJ	8	5	6
JB	6	3	9

6. ANALYSIS

As testing was limited to two weeks, the quality of the collected data must be taken as exploratory. Despite this, due to the high recorded impact on services both in surveys and general feedback, as well as the increasing documented access rate increasing over time, we can reasonably conclude a general positive impact of integrated documentation. Undocumented accesses can be attributed to forgetful users, partial services, service-training, and testing purposes. With managerial reinforcement and more experienced users we expect to see these rates continue to increase. We have linked the low usability with poor querying tool support, and we expect that better search capabilities will dramatically increase the effectiveness of the system as a whole. Despite that, the base functionality of our software has proven effective in document generation and our users have equivalently found

[1] https://github.com/draios/sysdig

generated documentation less intrusive and more useful than traditional documentation methods.

7. CONCLUSION

We are encouraged by the positive reactions to our software. Despite criticisms, the basic functionality of our system has proven robust. Instead of relying on users to generate documentation of their own volition, voluntarily interrupting their work flow, an integrated system has streamlined the process, generating more quality documentation than traditional methods, and While the software has proven beneficial, there are still a number of areas we can improve. The querying tool is the current major bottleneck, with migration to a single, authenticated, web portal for database searching. We would also like to support docx file extraction to contain documentation, user commands, and pertinent meta-data. As documentation becomes more consistent and accurate, we would like to begin tracking time to perform specific services to begin true quantitative impact analysis. Whilst our hypothesis has been proven, we look forward to continuing testing on this software.

8. ACKNOWLEDGMENTS

This project could not have been completed without the help of numerous individuals. My thanks to Dr. Cichanowski for networking and cryptographic advice, Dr. Iyengar for his unrelenting support throughout the research process, the head of my department, Ali Omar, for allowing me to test and implement on the university servers, my coworkers Jared Baudoin, Steven Granquist, and Travis Jurjevic for their testing, support, and advice throughout this project, and Grant, for believing in me every step of the way.

9. REFERENCES

[1] G. Aceto, A. Botta, W. D. Donato, and A. Pescapè, “Cloud monitoring: A survey,” Computer Networks, vol. 57, no. 9, pp. 2093–2115, Jun. 2013.

[2] K. Fatema, V. C. Emeakaroha, P. D. Healy, J. P. Morrison, and T. Lynn, “A survey of Cloud monitoring tools: Taxonomy, capabilities and objectives,” Journal of Parallel and Distributed Computing, vol. 74, no. 10, pp. 2918–2933, Oct. 2014.

[3] V. Phoha, “A standard for software documentation,” Computer, vol. 30, no. 10, pp. 97–98, Oct. 1997.

Evaluation of JavaScript Photo Animation Applications.

Sean Joyce
Winona State University
175 W Mark St,
Winona MN 55987
hp3513kj@go.minnstate.edu

ABSTRACT

Graphics Interchange Format (GIF) is an image format used to create and display a series of images to form an animation. GIF displays are used via memes, social media, texts, and emails. The popularity of this format has contributed to a surge of online JavaScript based animation applications.

We present an evaluation of JavaScript photo animation applications that can ascertain the most usable, effective, and strongest applications. The evaluation is designed to test each aspect of several JavaScript applications to fully determine the strengths and weaknesses. Aspects used for evaluation included execution time, photo quantity limit, photo quality limit, file support, downloaded file types, usability, cosmetics, and animation quality. The results were analyzed to identify the strong and weak software with respect to these aspects.

Initial experiments conducted using a variety of images suggests that there may not be any strong differences between the applications in terms of execution time, or animation quality. There seems to be significant differences in usability, and file support.

1. INTRODUCTION

JavaScript photo animation applications have transformed photo sharing, education, and social media. The Graphics Interchange Format (GIF) is a widely used image format for photo animation and the success of GIF is partially due to the many free to use JavaScript photo animation applications found online. These applications allow a user to upload a set number of images to create a small animation. The .gif file type allows the user to post or save the animation with ease. The variety of uses of photo animations are extensive, including education, entertainment, and research.

Proceedings of the 19th Winona Computer Science Undergraduate Research Seminar, May 1, 2019, Winona, MN, US.

1.1 Education

The use of .gif images within education enhances how students learn. Goudsouzian [1] found that animating historical figures with short stories about their past helped students retain more of the lesson [1]. By adding a dimension to the image, the students could see past the photograph and imagine the person. This benefit to memory recall has potential in all departments of education.

1.2 Entertainment

According to Alex Konrad a writer for Forbes magazine, GIPHY.com, an application tested in this evaluation, passed 100 Million Daily Users in late 2016. Each of these users contributed to the 1 Billion GIFS sent each day [5]. The .gif format has become a staple in social media and entertainment with most major instant messaging services not only allowing GIFs, but offering free to access GIF repositories.

1.3 Research

Similar to how GIFs assist students with learning, the .gif format also benefits researchers who need to convey the meaning or their work in a concise manner. Morgan [2] wrote in depth about how the use of GIFs was “ideally suited” [2] for archaeology outreach and experimentation, due the ability to share vital information without the audience knowing the specific terminology.

2. PROBLEM STATEMENT/ HYPOTHESIS

Any individual looking to create time-based animations finds that there are an extensive variety of software solutions online. If an individual is only animating a small number of photos any of these options should suffice, although if an individual is dealing with a large quantity of images or potentially high-resolution images certain software could underperform. There are many blog posts and social media exerts on which JavaScript software is the best although there are few research based evaluations. Which is the best free JavaScript based photo animation software currently online, and could a better design be implemented.

3. METHODOLOGY

3.1 Images

The materials used for this evaluation were 10 sets of images (Table 1). To ensure each software was tested against multiple variables 3 different image types were used; .JPG, .CR2(Raw Image), and .PNG. 8 sets of JPG images were used due to their high popularity. CR2 images are common among photographers but rarely used elsewhere, are very large in size and they were not

expected to be accepted so only 1 set of .CR2 images were used. A single set of .PNG images was tested to see if the transparent background offered by the .PNG format would be accepted.

Table 1 Image Sets

Image Set:	Image Count:	Image Size:	Image File Type:
1	24	6 MB	.JPG
2	43	6 MB	.JPG
3	23	6 MB	.JPG
4	14	4 MB	.JPG
5	22	5 MB	.JPG
6	25	5 MB	.JPG
7	113	6 MB	.JPG
8	136	2 MB	.PNG
9	34	6 MB	.JPG
10	16	26 MB	.CR2

3.2 Software

Eight different free to use JavaScript application were tested. These software applications were chosen using Google's search algorithm under the assumption the top search results would correlate with the most popular applications. Each software was graded using a grading matrix (Table 3). The grading considered both the product and the process. The animation quality, execution time and photo quantity limit were all graded as well as website usability and features (Table 2).

Table 2 Software Grading Matrix.

Image Set:	1	2	3	4	5	6	7	8	9	10
Start Time:										
End Time:										
Execution Time:										
Animation Size:										
Frames:										
Animation Quality										
Usability:										

Table 3 Software Detail Matrix.

Number:
Name:
URL:
Login Required:
Free to Use:
Adblock allowed:
Allows Videos:
Upload During Execution
Download Options:
Size Limit
Length(secs) limit:

3.3 Testing

The application testing was completed on a PC operating Windows 10, utilizing the Firefox browser. An animation for each of the 10 image sets were attempted on each of the 8 software applications. The start time and end time were marked using a mouse listener written in Python. The animation quality and usability were graded by the tester on a scale from 1-10. 1 being poor, 10 being perfect. For the applications that could not support the full count of images in the set, images were removed until the animation could be completed.

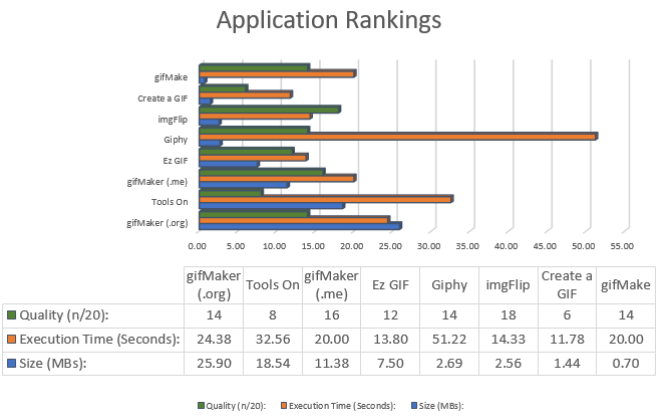
4. RESULTS

Initially we held the idea that there would not be any significant variance between the software applications and their output. Post evaluation we found quite the opposite. As shown in figure 4, comparing only the metrics animation size, and execution time one can already suggest that there are significant differences between the applications.

Certain software applications stood out from the rest in both positive and negative ways. The software provided by gifmake.com has a file size limit of 4MB, so only image set 8 successfully created an animation. Additionally, only the applications offered by Gifmaker.me, Gifmaker.org, Imgflip, and Create a GIF were compatible for the full image counts, the others forced the user to remove many photos.

4.1 Execution Time

Of the quantitative data, execution time was the most uniform with most of the applications completing on average within 10-25 seconds. Giphy.com was a strong outlier to the others in terms of execution time because the image upload was included in the GIF creation, slowing down execution time significantly.



	gifMaker (.org)	Tools On	gifMaker (.me)	Ez GIF	Giphy	imgFlip	Create a GIF	gifMake
Quality (n/20):	14	8	16	12	14	18	6	14
Execution Time (Seconds):	24.38	32.56	20.00	13.80	51.22	14.33	11.78	20.00
Size (MBs):	25.90	18.54	11.38	7.50	2.69	2.56	1.44	0.70

Figure 1 Application Rankings.

Table 4 Average Execution Time.

Name:	Execution Time (Seconds):
gifMaker (.org)	24.38
Tools On	32.56
gifMaker (.me)	20.00
Ez GIF	13.80
Giphy	51.22
imgFlip	14.33
Create a GIF	11.78
gifMake	20.00

4.2 Animation Size

Animation size was the most unique. The average animation size between the applications was close in comparison, with most of applications average output size being <10MBs. The variance comes from the individual animations sizes, which have a size range from 500KBs to 96MBs.

Table 5 Average Animation Size.

Name:	Size (MBs):
gifMaker (.org)	25.90
Tools On	18.54
gifMaker (.me)	11.38
Ez GIF	7.50
Giphy	2.69
imgFlip	2.56
Create a GIF	1.44
gifMake	0.70

5. CONCLUSION

The evaluation of JavaScript Photo Animation applications we conducted exceeded our expectations specifically in terms of variety. ImgFlip placed highest in this evaluation with a small file size, reasonable execution speed, and above average animation quality. Create a GIF placed the lowest overall due to the animation quality. Collectively the applications performed very well, with 7/8 successfully completing 90% of the animations. Future testing could be conducted to compare the free to use software applications tested in this evaluation to paid applications. Additionally, more photo sizes and file types could be tested. Any individual who wants to create a photo animation using an online software has many good choices, which is the best is specific to their needs.

6. ACKNOWLEDGMENTS

Special thanks to Dr. Shimin Li, Dr. Sudharsan Iyengar, and Dr. Nina Marhamati for their invaluable assistance throughout the research process.

7. REFERENCES

[1] Goudsouzian L. 2017. Photo animation brings scientists back to life in the classroom†. J. Microbiol. Biol. Educ. 18(1): doi:10.1128/jmbe.v18i1.1228

[2] Morgan, C. and Scholma-Mason, N. 2017 Animated GIFs as Expressive Visual Narratives and Expository Devices in Archaeology, Internet Archaeology 44. https://doi.org/10.11141/ia.44.11

[3] EURASIP Journal on Advances in Signal Processing2005**2005**:192492

[4] Sagar, K. & Saha, A. Hum. Cent. Comput. Inf. Sci. (2017) 7: 29. https://doi.org/10.1186/s13673-017-0111-8

[5] Konrad, Alex. "Giphy Passes 100 Million Daily Users Who Send 1 Billion GIFs Each Day, Reveals GV As Investor." *Forbes*, Forbes Magazine, 27 Oct. 2016, www.forbes.com/sites/alexkonrad/2016/10/26/giphy-passes-100-million-users-reveals-gv-as-investor/#441de12c4d64.

Using the Fourier Transform and Harmonics to Recreate Instrument Tonality

Cameron Pelzer
Department of Computer Science
Winona State University
CPelzer13@winona.edu

ABSTRACT
Each musical instrument makes distinct sounds, caused by harmonics particular to that instrument. Harmonics are the waves produced by an instrument that aren't the base pitch. By altering the harmonics around a target frequency, we can digitally produce sounds similar to that of various instruments. The first step in doing so is to understand the math behind the separation of the harmonics. The function known as The Fourier Transform uses an input signal to output data that can display graphically the breakdown of the signal's harmonic frequencies. This data can then be stored as a template for the breakdown of different instrument's sound. The application outlined in this paper was created to build, save, and use these templates. Using this application, templates for Trumpet and Violin were created by saving the ratio of the base note to the harmonics. Using this template and the inverse Fourier Transform, the application can alter a simple string of notes to sound like the templated instrument.

General Terms
Algorithms, Human Factors

Keywords
Spectral Analysis, C#, Timbre, Harmonics, Fourier Transform

1. INTRODUCTION
1.1 Background Information
Sound is made up of several elements: pitch, amplitude, and timbre. Pitch is determined by the base frequency that is being omitted by the instrument. Amplitude is how loud and instrument is playing a frequency. Both pitch and amplitude are relatively easy to measure. To measure pitch, you analyze a sound wave and find the most prominent frequency. To measure amplitude, you analyze a sound wave and look for the amplitude of the wave itself. Timbre is the hardest to measure. Timbre is a combination of several different harmonic frequencies that are being omitted from an instrument when a base frequency is played. The different amplitudes of these harmonic frequencies are what gives different instruments their timbre. This timbre is going to be the main focus of this paper. To measure timbre, one must use a function known as the Fourier Transform. This function is then used to make a power spectrum analysis. Figures 1 and 2 show two examples of these graphs.

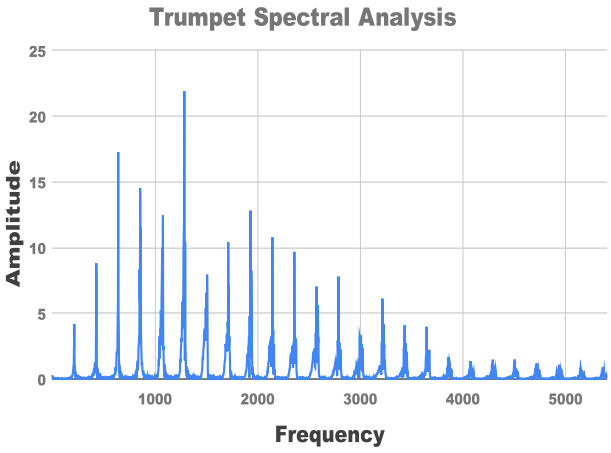


Figure 1. Power Spectral Analysis Graph for a Trumpet

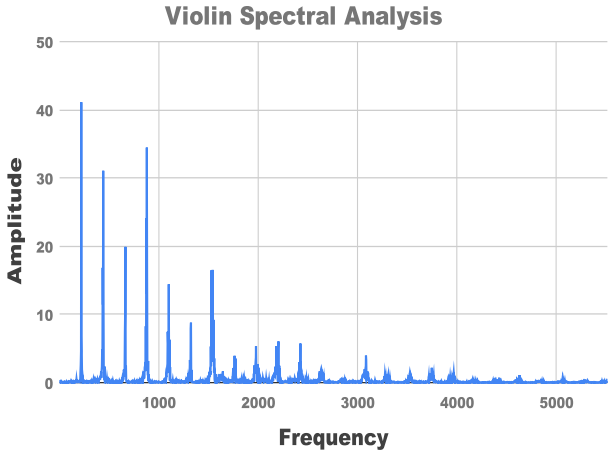


Figure 2. Power Spectral Analysis Graph for a Violin

Proceedings of the 19th Winona Computer Science Undergraduate Research Seminar, May 1, 2019, Winona, MN, US.

These graphs look at the frequency and amplitude of each harmonic given off by the instruments. The first spike in the graph is the pitch you hear. The other spikes in the graph are what make up the timbre. The differences in the amplitude of the harmonics in the above graphs is a visual representation of the differences in the instrument's timbre.

1.2 Problem Statement

To create a program that can decompose an audio wave, save the levels of the harmonics as a template, then compose a wave with the template on top of a new base note to create a synthesized sound that mimics that of the instrument saved in the template.

1.3 Motivation

Musical production software gives artists several ways to record musical input. Artists can play notes directly into a microphone, they can feed the software MIDI input from a keyboard, or they can manually click and select each note in the software. MIDI input and manually clicking notes makes use of software plugins to simulate different instruments. These simulated instruments generally have a sound byte of that instrument playing each note. This allows for fairly accurate simulation of the instrument without having to play the instrument. This means that artists need to have basic piano knowledge in order to use MIDI input efficiently. If an artist doesn't have any piano knowledge, they must manually click out each note in their production software. This project aims to reduce the effort of non-technical artists so they may be able to produce quality music using different instruments. This software, in its completed form, would allow users to sing into a microphone and lay any instrument template over it to create a string of notes that sounds like the templated instrument. This gives users a new way to record musical input that doesn't require as much effort as manually clicking in each note and it doesn't require as much technical skill as playing an instrument or using MIDI input.

2. METHODOLOGY

2.1 Audio Data

To create a program that decomposes a wave, an understanding of how audio data is stored in important. There are several different types of audio files: MP3, WAV, MP4, etc… These files all store audio data in a similar way. They represent a wave by using a positive and negative scale to show a wave's current position with reference to time [2]. Some audio files use compression to save space, but in order to create accurate templates, we want to use a file type that doesn't use compression. This file type is the WAV file. Sapp shows [6] the structure of the WAV file. This includes the different fields that make up the header, followed by the raw data. WAV files store the raw data as sixteen-bit two's compliment integers. It can store either one channel for a mono recording, or two channels for a stereo recording. The header also contains data regarding the sample rate, and the number of channels which will be useful when it comes time to running the data through the Fourier Transform.

2.2 Sampling Audio

The next step is to obtain samples of the instrument sound. Using a program called FL Studio, clean samples of a Violin and Trumpet, playing an A3 with a frequency of 220 Hz, were obtained. Each sample was 2 seconds long, and was sampled using a 44,100 Hz sample rate. This means that 44,100 times a second, the position of the sound wave is being recorded. The recording was also made using mono sound. This is for simplicity sake. If stereo sound was recorded, the data would have to be separated into the left and right channels. Each channel would then be run through the Fourier Transform, and the results of each channel would then be combined to show the overall spectral analysis of the recording.

2.3 Fourier Transform

The Fourier Transform for discrete functions is as follows [8]:

$$X_k = \sum_{n=0}^{N-1} x_n e^{\frac{-i2\pi kn}{N}} \quad \mathrm{K} = 0, 1, \dots, \mathrm{N} - 1$$

This equation takes in real numbers, and outputs a complex number. This function was created to separate waves into their relative frequencies. It's mainly used in physics and electronic engineering to analyze the frequency of circuits [1]. The Discrete Fourier Transform sums up the relative presence of the different frequency of waves [3]. This means the more present a frequency of wave is in the sample, the greater the output will be at that frequency. Using this, we can break down an audio sample into its base pitch and relative harmonics. Now that there is an understanding of the mathematics, the code can be created.

2.4 Data Manipulation

Using C#, the next step is to build software to decompose the wave. Pulling all the data into an array allows for the program to have indexed access to the data points. WAV files use 2's compliment signed integers to record the data. Before the data can be passed to the Fourier Transform, it must be converted to a double, and normalized. To do this, the header of the WAV file is skipped and the 16 bit 2's compliment integers are saved into a temporary variable and normalized by dividing by the maximum value, 32,768. Once the values are normalized, they are saved into an array. This array can now be passed to the Fourier Transform for processing. Oakley [4] created an algorithm using 3 classes in C# to complete the Fourier Transform. Using this algorithm, an array of real numbers was given as input, and an array of complex numbers was given as output. By using that array of output, another array can be created that computes the magnitude of the relative presence from the complex output.

Using this array of complex outputs, an array can be created that holds only the important values. These include all values above a magnitude of 3, and their frequencies. This array then contains all the frequencies and magnitudes that make up that particular instrument's timbre at a particular base frequency. By dividing all frequencies by the base frequency, and dividing all magnitudes by the base magnitude, each value can be saved as a ratio to the base frequency. This means that now, this template can be multiplied by any base frequency to create an array consisting of the timbre of the instrument, but layered on top of the new base frequency. Once this array is created, it can be run through the Inverse Fourier Transform to create an array that holds the wave data in relation to time.

2.5 Inverse Fourier Transform

The Inverse Fourier Transform is as follows:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{\frac{i2\pi kn}{N}} \quad \mathrm{n} = 0, 1, \dots, \mathrm{N} - 1$$

This function creates a complex sine wave for each frequency. It then multiplies each wave by its Fourier coefficient, then it sums all the waves together. The sum of the waves is then divided by the number of waves. The result is one wave that contains all the frequencies of the templated instrument relative to the new base frequency. This new wave will sound like the templated instrument, but it will not have any attack or decay. It will be a constant wave.

3. CONCLUSION

Using one sample of audio from an instrument, the timbre can be recreated and applied to several different base frequencies. Using the Fourier Transform, a wave was able to be decomposed. This wave was then filtered down to only the important values. These important values were then made into ratios by dividing them by the base frequency and magnitude. This allowed for the recreation of the timbre for any base frequency. Then using a second input file, a new base frequency is obtained. The instrument template is then laid on top of this frequency to create a new sound. Using the Inverse Fourier Transform, a new wave is constructed using the new base frequency and the instrument template. This creates a sound that could be accurate given it is modified to have an attack and decay. The output of the overall program is a new WAV file that contains the audio using the new base frequency and the instrument template. This program is limited to only single note for input, and a single note for output.

4. FUTURE WORK

In the future this program would be extended to having full song output. This means that it could be applied to cover a full string of notes. Once that is implemented, the program could be modified to be used as a plugin for musical software. This would allow artists the ability to sing a string of notes into the program and have that string of notes converted into an instrument. Doing so gives artists who doesn't play instruments the ability to record their own music much faster than using manual note input. This could potentially open up the production market to a whole new group of artists who had the idea of what the music would sound like, but no way to put that music into a tangible form.

5. ACKNOWLEDGMENTS

Special thanks to Dr. Barry Peratt of the Mathematics Department at Winona State University for the contribution to this project. Thanks also to Dr. Sudharsan Iyengar and Dr. Nina Marhamati of the Computer Science Department at Winona State University for their contributions to this project.

6. REFERENCES

[1] An Interactive Guide To The Fourier Transform. (n.d.). Retrieved April 1, 2019, from https://betterexplained.com/articles/an-interactive-guide-to-the-fourier-transform/

[2] Dobrian, C. (1997, December). *Digital Audio*[Scholarly project]. Retrieved April 1, 2019, from http://music.arts.uci.edu/dobrian/digitalaudio.htm

[3] Kundur, D. "Magnitude and Phase." Magnitude and Phase. www.comm.utoronto.ca/~dkundur/course_info/signals/notes/Kundur_FourierMagPhase.pdf

[4] Oakley, C. (2011, November 15). *Development of a Fourier transform in C#*[Scholarly project]. Retrieved April 1, 2019, from https://www.egr.msu.edu/classes/ece480/capstone/fall11/group06/style/Application_Note_ChrisOakley.pdf

[5] Rose, N. & Holloway, D. (2012, November 23). *Finite element modeling of brass musical instruments*[Scholarly project]. Retrieved April 1, 2019, from https://www.logosfoundation.org/instrum_gwr/bug/p60.pdf

[6] Sapp, C. "WAVE PCM Soundfile Format." *Microsoft WAVE Soundfile Format*, www.soundfile.sapp.org/doc/WaveFormat/

[7] Stoica, P. & Moses, R. (2004, February 1). *Spectral Analysis of Signals*[Scholarly project]. Retrieved April 1, 2019, from http://user.it.uu.se/~ps/SAS-new.pdf

[8] Weisstein, E. W. (n.d.). Fourier Transform. Retrieved April 1, 2019, from http://mathworld.wolfram.com/FourierTransform.html