



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

How can we mine cryptocurrency using available computing capacity in the dark web/deep web by Distributed Network Attack (DNA)?

by

Jordan Phillips

Supervisor: Dr Carlene Campbell

Second Marker: Dr Kapilan Radhakrishnan

**Project submitted as part of the requirements for the award of
MSc Computer Networks and Security**

September 2019

Declaration of Originality

I, Jordan Phillips, declare that I am the sole author of this Project; that all references cited have been consulted; that I have conducted all work of which this is a record, and that the finished work lies within the prescribed word limits.

This work has not previously been accepted as part of any other degree submission.

Signed:



Date: 30/9/2019

FORM OF CONSENT

I, Jordan Phillips, hereby consent that my Project, submitted in candidature for the M.Sc. Computer Networks and Security degree, if successful, may be made available for inter-library loan or photocopying (subject to the law of copyright), and that the title and abstract may be made available to outside organisations.

Signed:



Date: 30/9/2019

Abstract - As technology continues to advance, so too does the need for greater levels of security. Network and cyber security concepts are based upon the three pillars of security, namely: confidentiality, integrity and availability. Confidentiality refers to ensuring outside parties do not gain access to sensitive or personal information. Integrity ensures that data is not tampered with, and, therefore, its content is not altered in any capacity. Finally, availability refers to the insurance that data may be accessed by authorised groups or individuals at any time, or from any location. Two key fields of data protection are Cryptography and Cryptanalysis.

Cryptography is the study of altering data into an unreadable format, known as ciphertext, which prevents unauthorised parties from accessing sensitive and / or classified information. Specialised codes, known as keys, are utilised in retrieving the original plaintext data from an encrypted ciphertext. Cryptanalysis is the study of deciphering encoded messages without any knowledge of the key. The goal of cryptanalysts is to decrypt ciphertexts without any knowledge of the plaintext, encryption key or the algorithm used in encryption. Many forms of encryption exist, with cryptography and cryptanalysis specialists constantly improving upon their designs.

The aim of this project was to test and identify potential weaknesses within modern ciphers. As such, this project opted to utilise Advanced Encryption Standard (AES) and Rivest–Shamir–Adleman (RSA) as examples of encryption ciphers. This decision was based on research into these ciphers, namely, their high levels of security and widespread usage within modern technological infrastructures. Utilising these two ciphers also allowed this project to analyse a symmetric and an asymmetric cipher design, allowing for greater variance in research and testing.

Many software exists in order to analyse different ciphers. This project opted to utilise three main bits of software: Distributed Network Attack (DNA); CrypTool 2; and Kali Linux, which was run utilising Oracle VM software. DNA is a newer form of password recovery and decryption software, developed by AccessData. It functions through allocating various tasks to different devices spread throughout a network. As such, this software utilises unused computing capacity of devices to work together in completion of a single decryption task. CrypTool is a cryptography and cryptanalysis software that possesses several tools for encryption and decryption purposes. Finally, Kali Linux is an Operation System (OS) which can be used to launch numerous scans and attacks, typically for penetration testing purposes. As it was identified that newer ciphers are resilient to standard cryptanalysis attacks, this project opted to explore other avenues of attack in conjunction with previously mentioned software.

Certain vulnerabilities were identified using this combination of software. Examples of these include side-channel attacks, Heartbleed exploitation attacks, differential attack and BEAST and POODLE attacks. These attacks target vulnerabilities within a cipher's implementation, rather than within a cipher itself. As results shown from testing using DNA and CrypTool showed high levels of resistance to standard cryptanalysis attacks, such as known plaintext, chosen plaintext, ciphertext only and differential analysis, exploitation of implementation vulnerabilities was identified as one of the few true weaknesses of AES and RSA ciphers.

Keywords – Cryptography, Cryptanalysis, Kali Linux, CrypTool, Distributed Network Attack

Table of Contents

Chapter 1	6
1.1 Introduction.....	6
1.2 Aims.....	8
1.3 Objectives.....	8
Chapter 2	9
2.1 Literature Review	9
2.1.1 Security	9
2.1.2 Cryptography	13
2.1.3 Cryptanalysis.....	16
2.1.4 Distributed Network Attacks (DNA).....	19
2.2 Related Work	19
Chapter 3	22
3.1 Methodology	22
3.1.1 Philosophies	22
3.1.2 Approach	22
3.1.3 Strategies.....	23
3.1.4 Choices	23
3.1.5 Time Horizons	24
3.1.6 Techniques and Procedures	24
3.1.7 Risk Assessment	24
Chapter 4	25
4.1 Design.....	25
4.2 Implementation	27
4.2.1 Kali Implementation.....	28
4.2.2 CrypTool 2 Implementation.....	31
4.2.3 Distributed Network Attack (DNA) Implementation	37
Chapter 5	38
5.1 Results.....	38
Chapter 6	46
6.1 Conclusion	46
6.2 Future Work.....	48
References.....	50

Table of Figures

Figure 1: DNA Layout.....	25
Figure 2: CrypTool 2	26
Figure 3: Kali Linux OS	27
Figure 4: Heartbleed Vulnerability Scan.....	28
Figure 5: Heartbleed Vulnerability Scan Output [101].....	28
Figure 6: Heartbleed Vulnerability Scan Alternate Output	29
Figure 7: Metasploit Within Kali	29
Figure 8: Metasploit Heartbleed Options	30
Figure 9: Metasploit Heartbleed Results [101]	30
Figure 10: AES Encryption Process	31
Figure 11: AES Ciphertext.....	31
Figure 12: AES Decryption	32
Figure 13: No Known Key-bits	32
Figure 14: Test 1 Display.....	33
Figure 15: RSA 256-bit Key Generation.....	34
Figure 16: JavaScript decoder (256-bit key)	35
Figure 17: RSA Encryption.....	35
Figure 18: RSA 512-bit Key Generation.....	36
Figure 19: JavaScript Decoder (512-bit key).....	36
Figure 20: Factoring Design.....	36
Figure 21: First AES Test Run Timer.....	38
Figure 22: First AES Test Completion.....	39
Figure 23: Second AES Test Run Timer.....	39
Figure 24: Third AES Test Run Timer	40
Figure 25: AES Second Test Completion.....	40
Figure 26: AES Second Test Completion Time.....	41
Figure 27: AES Fourth Test.....	42
Figure 28: AES Fifth Test	42
Figure 29: Factoring N Value of RSA 256-bit Key	43
Figure 30: RSA 256-bit Key p and q Values	44
Figure 31: RSA 256-bit Decryption.....	45

Chapter 1

1.1 Introduction

The internet is constantly evolving and has become a key component in everyday life, company infrastructure, general healthcare and even military operations [1] [2] [3]. These systems provide improved efficiency, higher quality of life for people all over and improved communication capabilities for both businesses and consumers [1] [3]. However, as networks continually advance, and more individuals and companies utilise these networks, the number of risks posed by various security threats increases [1] [3] [4]. Any and all devices and networks with internet connectivity are vulnerable to numerous threats, such as data leakage and data loss [1] [2] [3] [4]. This means that network security managers must consider any threat that can compromise network performance and data integrity [1] [3] [4].

Data loss and data leakage affect multiple types of network design, such as hierarchical and cloud [1] [3] [5]. Network security managers can take steps to mitigate these threats, such as knowing who is accessing what within the network and limiting user access based on specific criteria and traits [1] [3]. An example for limiting within cloud networks would be to limit user access based on location or device used to access the network [5]. A method for detecting data leakage can be done utilising watermarking [6]. This method embeds a unique code in each distribution copy of data [6]. Through this, if information is discovered in the possession of unauthorised individuals or groups, it is possible to identify the leaker [6]. It should be noted however, that these watermarks can be destroyed if the data recipient is malicious [6].

Data security models are based on cryptography theories, which research data confidentiality, integrity and availability, among other data security defence strategies [1] [3] [7]. Confidentiality ensures that unauthorised individuals or groups are unable to access information [8] [9]. This ensures only those authorised may access sensitive data [8] [9]. Should data become available to individuals that are not authorised to view this information, whether intentionally or by accident, then data confidentiality has failed [8] [9]. This failure is commonly referred to as a data breach [10]. These breaches have massive negative implications for businesses, organisations and customers or clients involved [10]. Personal information are common forms of data breaches, such as identification, addresses, phone numbers and bank information [10]. These are some of the most critical pieces of information regarding individuals and are the most valuable for those with malicious intent [10].

Data integrity refers to the authenticity of information [8] [9]. This means ensuring that information is not altered from the original content and that the source is legitimate [8] [9]. Therefore, the data integrity process focuses on ensuring data validity and accuracy within databases and during their lifespan [8] [9]. Ensuring data integrity is key, as it allows for data recoverability and search ability, data traceability and connectivity [8] [9]. Data integrity can become compromised through human error, both maliciously or by accident; transfer errors, where data can come altered during transfers between devices; viruses and malware; and faulty or tampered hardware [8] [9].

Finally, data availability refers to the accessibility of data by authorised users [11]. Typically, data availability is achieved through redundancy [11]. In this scenario, redundancy refers to additional storage and paths to reach data, ensuring that it is accessible constantly and, in the event, specific areas become compromised [11]. Data is typically accessed over high-speed connections, such as fibre optic, allowing for quick access to stored data [11]. Cloud is one

method of storing data, which has the benefit of additional backups of stored data, further enhancing data availability [11]. In the event malicious individuals or groups fail to breach the previous stages of data security, availability becomes a target. A common attack in preventing data availability is the Denial-of-Service (DoS) attack [12]. DoS attacks target availability of a system resources by flooding servers, systems or networks with traffic with the aim of overwhelming the system [12]. This causes legitimate users to become unable to access system resources, nullifying data availability [12].

Cryptography is the process of converting standard texts into an unreadable form [7] [13] [14]. The purpose of this is to prevent unauthorised people from accessing sensitive information [7] [13]. When developing cryptography algorithms, the developer must consider data confidentiality, integrity and availability [13]. Therefore, in order to maintain all three features of data security, special keys known as secret keys were used to convert data from plaintext to cypher text, or cypher text back into plaintext [13] [14]. Before communication can begin, both sides must agree upon protocols and how they are to exchange information, such as key specifics [13] [14]. Cryptography can be classified into two types based on requirements and applications [13] [14]. These are Symmetric Key Cryptography and Asymmetric Key Cryptography [13]. Symmetric key utilises the same key for both encryption and decryption purposes where Asymmetric utilises two variant keys, one for encryption and one for decryption [13].

Cryptography is one key aspect this project will focus on, and the different areas and methods within this field will be examined in detail within the literature review located in chapter 2 of this paper. Other fields which are related to the project topic include the Deep / Dark Web and Cryptocurrency.

The Dark Web is part of the World Wide Web that requires specific software in order to access [15] [16]. There are specific characteristics associated with the dark web that differentiate it from the standard web, such as:

- Decentralisation [16]. Peer-to-peer is the most common method of communication used in the dark web [16].
- Internet Infrastructure Utilisation [16]. The dark web is built upon the standard public internet infrastructure [16].
- Utilising non-standard computer protocols [16]. Both communication protocols and ports used are unreachable or unusable by those outside of the network [16].

The dark web ensures users can only access contents using specific protocols and that a user's identity cannot be traced back [15] [16]. Protocols which can be used to access the dark web include Tor (The Onion Router), Onion, I2P, Freenet and IPVanish [15] [16]. These protocols are available to the public, with Tor being one of the more common protocols used [16].

Cryptocurrencies are a type of currency that exist only within the digital world [17] [18]. The field of cryptocurrency has grown rapidly since the development of bitcoin back in 2009 [17]. These currencies do not require third parties and are transferred from person to person [17]. The elimination of third parties also eliminates the pay requirement associated with their services [17] [18]. Cryptocurrencies are based on blockchain, an append-only ledger that users can access using certain applications known as wallets and exchanges [17] [18]. Blockchains

are series of data blocks that create a data chain, where later blocks refer to a single, previous, block located within the chain which is identified with its hash [17] [18].

1.2 Aims

The overall aim of the project is to identify if it is possible to mine cryptocurrency using available computing capacity in the dark web/deep web by Distributed Network Attack (DNA). However, due to time constraints regarding the project, it was decided that this overall goal is not possible. Therefore, this project will focus on one key aspect, namely the cryptography section.

As such, the aim for this paper is as follows:

- To analyse the effectiveness of various encryption methods by utilising cryptographic / cryptanalytic software such as Distributed Network Attack (DNA).

1.3 Objectives

To accomplish this goal, several steps will be taken. The following objectives have been identified which will aid in the completion of the project aim:

- Research into the field of cryptography. This objective is vital as there are various forms of encryption. Therefore, understanding the underlying process involved and how various aspects of encryption and decryption function will play a vital role in project success.
- Research into cryptanalysis. This field is the study of acquiring the plaintext data back from encrypted ciphertext without possessing the key or knowledge of the algorithm.
- Identify encryption algorithms to be used in various tests in order to determine their security strength levels. There are many encryption algorithms utilised. Therefore, it is important to research them in order to determine which ciphers are more suitable for testing purposes.
- Research into cryptanalysis attacks and other potential vulnerabilities of chosen encryption ciphers. It is important to understand how to perform attacks in order to obtain accurate data.
- Design experiments utilising different encryption ciphers to be tested against various cryptanalysis attacks. Experiments must be able to be set up and run utilising software and hardware readily available for the scope of this project.
- Analysing results to determine the efficiency of different encryption ciphers. The results obtained from various experiments will aid in determining which cipher possesses greater security level, as well as determining the main vulnerabilities chosen ciphers possess.

The result of these objectives should determine the strengths of various ciphers, as well as potential methods of circumventing their security in order to retrieve the original plaintext. These tests should also determine effectiveness of software utilised, including Distributed Network Attack (DNA).

Chapter 2

2.1 Literature Review

This report has touched upon the concepts of data confidentiality, integrity and availability and their role within data security. These concepts are also critical to consider when dealing in the field of cryptography [13]. In order to gain a greater understanding of the field associated with this project, research into various aspects of internet and network security, cryptography, cryptocurrency and the dark/deep web is advised. Primarily, the author will focus on gathering information related to encryption methodology and their role within the overall field of security. From here, analysis into decryption methods and software will provide useful information for the completion of the project aim.

2.1.1 Security

Network security is the field of study which prevents and protects networks against unauthorised intrusion from individuals or groups of people [3] [19]. This field has been noted as some as becoming the fifth security field after sea, land, air and space [3]. Recently, a great deal of research has been conducted within this field, with primary developments being related to data security and network security [3]. Basic definitions of network security are to protect, where a network is designed to prevent unauthorised access to systems; detect, the ability to identify configuration changes or traffic which would indicate network problems; and react, the ability to respond to problems and return the network to a safe state as quickly as possible [3] [19].

Regarding implementing network security methods, relying on single means of protection is typically considered bad practise [3] [19] [20]. This is because a single defensive tool or software can be overcome by persistent attackers [3] [20]. There are multiple methods which can be employed to improve network security, including:

- Access Control [21] [22]. This relates to the ability to allow or deny user access based on pre-defined criteria [3] [21] [22]. One such method is to block communication based on ports and access points in order to secure communication between authenticated and authorised devices [21]. Access Control Lists (ACLs) are another security method utilised for user control [22]. This method is widely used in ISPs to carry out security access policies for each individual user [22] [23]. ACLs are an instruction list used to permit or deny packets from entering or exiting a network [22] [23]. Common privileges granted via ACLs include the ability to read a file, to write to files and to execute a file program [22] [23].
- Anti-malware software [24]. Malware has been increasing over the years, and many antivirus vendors have developed software to detect and protect systems from these threats [24]. However, there are some malware that can evade detection by this software and continue to pose a threat to systems and networks [24]. Most anti-virus software use signature-based detection to detect malware, but this can be an ineffective method regarding some malware [24]. An example of malware capable of avoiding these software's is Agobot, which possesses a built-in anti-debugging function designed to disrupt analytical investigations [24].

- Application security. Untrusted applications are common points of attack for attackers to gain access to a network. Therefore, ensuring only trusted applications are utilised is important in preventing unauthorised access to network data [3].
- Data loss prevention methods, such as backups which ensure data availability is maintained [11]. Other forms of data loss include human error, therefore, having policies and software in place to prevent accidental or deliberate exposure of data is important. Such methods include the previously mentioned ACLs [22] [23].
- Firewalls [25] [26]. These tools are used to protect a network from malicious activities by detecting and filtering packets that pass through the firewall itself [25] [26]. The security level depends on verification measures based on rules and policies enforced within the organisation [25] [26]. Firewalls check packets based on several criteria, such as source or destination IP address, source or destination port number and protocol types [25] [26]. Following this, the firewall will either accept, deny or discard the packet [25] [26]. Firewalls can be classified into two kinds of products: packet filtering firewalls and application firewalls [26].
- Intrusion detection [27] [28]. These systems provide dynamic protection to a network, based on web self-defence [27]. Intrusion detection technologies can detect both known intrusions and unknown intrusions [27] [28]. This refers to the capability of defending against both internal and external attacks [27] [28]. Recently, research on intrusion detection has gradually inclined to artificial intelligence technology as a means of improving detection accuracy [27].
- Wireless security [28] [29]. Wireless devices are just as susceptible to attacks as standard network devices and require steps to be taken in order to maintain data confidentiality, integrity and availability [29].
- Virtual Private Networks (VPNs) [30] [31]. This type of software is used to obtain secure and private communication and is a popular tool for online security [30] [31]. Traffic sent with the use of VPNs is encrypted and works by using shared public infrastructure [30] [31]. Privacy is maintained in this environment using tunnelling protocols [30] [31]. These “tunnels” can only be accessed using the proper encryption set by the VPN, i.e. traffic is encrypted at the source, sent across the public internet domain via tunnelling, and subsequently decrypted at the destination [30] [31]. Common VPNs utilised are IPsec and SSL [31].

As noted, there are many methods of ensuring network and data security. There are also methods which can be utilised to test network security, such as penetration testing [32] [33]. Two variations of penetration testing exist, namely black-box and white-box methods [32] [33]. White-box methods range from analysis to dynamic monitoring while black-box methods generate inputs and observe responses [32] [33]. Typically, black-box methods replicate the appearance of an external attacker [33]. As such, it is a useful method for testing network defences set to prevent attacks [33].

There are several attacks that target three pillars of network security, i.e. attacks that target data confidentiality, integrity or availability [12] [34] [35]. Attacks on these areas are typical forms of cybercrime [35]. Malicious groups can target these pillars to disrupt network operations; acquire sensitive data such as passwords, emails, bank details and other personal information; compromise and change data; and prevent legitimate users from accessing network resources [12] [34] [35]. Attacks that target data confidentiality include:

- Packet Capturing [36] [37] [38]. This form of network analysing and network attack is also known as packet sniffing [36] [37] [38]. One important feature in network security is monitoring packet over network connections [36] [37] [38]. This type of software operates by intercepting and logging network traffic utilising wired or wireless network interfaces on the host device [36] [37] [38]. Regarding wired networks, what can be intercepted depends on the network layout [37] [38]. On wireless networks, only one channel may be intercepted at a given time, unless the host device possesses several wireless interfaces that allow for multichannel capture [37] [38]. Network technicians can utilise information gained in order to troubleshoot areas of a network by determining which device fails to respond to network requests [36]. However, there are individuals who utilise this software for malicious purposes [36] [37] [38]. Hackers can use sniffers to eavesdrop on unencrypted traffic and capture information such as passwords and authentication tokens [37] [38]. This information can also be utilised to perform further attacks within a network [12] [37] [38]. Common packet sniffer software includes Wireshark and TCPDump [37] [38].
- Password Attacks, which are used to hack the passwords of users of a target device to gain access to the network [39]. While technology has advanced considerably, a common weak point remains the human factor [39]. Brute-force and dictionary attacks utilise this weakness to attempt to gain network access [39].
- Port Scanning is a type of network attack where the attacker attempts to discover services running on a target device by scanning TCP/UDP ports [40]. The goal of this attack is to determine which ports are in an open state [40]. From here, an attacker can determine which services and products are running on the victim device [40]. Vulnerabilities of these services can then be exploited to gain access to the device [40]. Nmap is one such tool that can be used to perform port scanning attacks [40].
- Ping Sweep is a method in which an attacker sends ping packets to a range of IP addresses to determine which ones respond with a reply [40]. This can be used by an attacker to determine which devices are up and which are down.
- Wiretapping and Keylogging are similar attacks in that both are methods of eavesdropping. Keylogging programs run in the background and log a user's keystrokes, which can be used to acquire passwords and other sensitive information. Wiretapping on the other hand involves hacking a telecommunication device to listen on phone calls of other users.
- Phishing, which is a method where attackers attempt to gain important information, such as passwords, credit card information and other personal details [41]. Firstly, an attacker obtains an email address of a victim [41]. From here the attacker generates a phished page – a copy of a legitimate page – that appears genuine but is in fact a fraud [41]. A link to this false page is then sent via email to a victim and should said victim believe that this site is legitimate, they may enter personal information into this site which the attacker has complete control over [41]. A method of preventing this form of attack is to not click on untrusted emails or emails sent from unknown sources [41].

Examples of network attacks that can compromise integrity are:

- Salami attacks. These are a series of small attacks that build up to a major attack [42]. Due to the nature of this attack, it can be difficult to detect [42]. As the attacker only

takes small amounts of data, victims of this attack remain unaware and therefore do not act [42]. As such the attacker can remain undetected [42].

- Data diddling attacks. This attack involves the illegal or unauthorised alteration of data within a database. The change typically occurs before or as the input is placed into a computer or output.
- Trust relationship attacks. The goal of this attack is to compromise a trusted host and subsequently use this host to launch further attacks on other hosts within a network [43]. If a trusted host exists outside of a firewall, that host can be used to attack a network through this firewall [43].
- Man-in-the-Middle (MitM) attacks [44] [45]. This attack refers to when a third party takes control of a communication channel between two or more parties [44] [45]. The attacker can intercept, modify, change or replace the targets communication traffic [44] [45]. Victims of this attack are unaware of the presence of a malicious third party, believing that the communication is well protected the entire time [44] [45]. This attack can allow an attacker to launch further attacks, such as Distributed Denial-of-Service (DDoS), DNS Spoofing, port stealing and session hijacking [44] [45]. Attackers can also obtain passwords, online IDs, telnet sessions etc. via this attack [44] [45]. These attacks can be launched in passive or active modes [44]. Passive attacks conceal the attacker's presence, meaning they go undetected but is only able to take data being sent between parties across a channel [44]. Active attacks, on the other hand, allow an attacker to intercept and manipulate data being sent to the destination [44]. As such, during passive attacks an attacker simply records information, whereas active attacks cause an attacker to modify information [44].
- Session hijacking attacks [46]. During this attack, an attacker will neutralise one device participating in an active connection [46]. The client machine is typically the target during these attacks [46]. From here, an attacker will take the place of the neutralised machine within the communication exchange [46]. Connection to the original device is typically terminated via Denial-of-Service (DoS) attacks [46]. A combination of DoS and MitM attacks are used to carry out session hijacking [46].

Finally, the following attacks are often used to disrupt data availability:

- DoS [12] [47] [48]. This attack attempts to disrupt network availability for users [12] [47] [48]. An attacker will launch many service requests which overload and consume all network resources, slowing down the network drastically or completely shutting it down [12] [47] [48]. In the event of server shutdown, due to the large number of requests, legitimate users will not be able to access the server, thus completely denying network availability [12] [47] [48]. This attack is also used in tandem with MitM attacks to perform Session Hijacking attacks [46]
- DDoS [47] [49]. This attack is a variation of a DoS attack, with the difference that the attack will originate from multiple, often millions of, IP addresses [47] [49]. The effects of this attack are like that of standard DoS attacks, in that the large quantity of requests overloads a server, bringing all services to a stop as the server is brought down [47] [49]. This, once again, completely denies all legitimate users from accessing network resources, denying networks availability [47] [49].
- SYN Flood attacks [50]. This attack is a major type of DDoS attack and is categorised as a semantic attack as it exploits a feature of TCP called three-way handshake [47]

[49] [50]. Typically, in a TCP connection process a client would send a TCP segment with the SYN flag set to 1, where a TCP server would reply with both SYN and ACK flags set to 1 [50]. Finally, the client would send a segment with ACK flag set to 1, initiating the connection process [50]. During the time between a client receiving a server SYN-ACK packet reply and acknowledging this connection, a server allocates space within a memory buffer to identify this half-open connection [50]. Therefore, during a SYN Flood attack, an attacker will send many SYN packets to a target without acknowledging the corresponding SYN-ACK packets [50]. As a result, this attack overflows the limited memory of the backlog, leading to legitimate SYN packets being discarded and the service becoming unavailable [50]. False SYN packets are difficult to differentiate from legitimate SYN packets, with the sole exception being the source IP address, which is spoofed when originating from an attacking client [50].

2.1.2 Cryptography

As previously mentioned, cryptography is a method of protecting data and communications by utilising codes which only authorised groups can process [7] [13] [14] [51]. This method prevents eavesdroppers from understanding information being sent across communication channels [7] [13] [14] [51]. The cryptography process involves converting plaintext into an unreadable code format, known as cypher text, in order to prevent unauthorised groups from acquiring the information [7] [13] [14] [51]. To convert cypher text back into its original structure, specific keys are required [7] [13] [14] [51]. These keys differ depending on the encryption method utilised and must be discussed between devices participating in encrypted communication [7] [13] [14]. There are multiple algorithms utilised in cryptography for various tasks [7] [52]. Encryption, message authentication, key exchange and decryption are some of the tasks which utilise various cryptography algorithms [7] [52].

There are two primary types of cryptography: Symmetric key cryptography and Asymmetric key cryptography [13] [52] [53]. Symmetric key systems utilise the same key for both encryption and decryption purposes [13] [52] [53]. Therefore, both source and destination systems are required to possess the key for communication to take place [13] [52] [53]. This form of encryption is also known as secret-key or shared key encryption [53]. The key needs to be shared via secret communication in order to prevent malicious parties from acquiring the key [52] [53]. As decryption processes utilise the same key as encryption processes, an attacker could easily reconstruct cypher text back into plaintext should they intercept the key prior to communication [51] [52] [53]. One advantage to this form of encryption is that it utilises less resources and is faster, due to only a single key utilised at both ends of communication [52] [53]. Various algorithms have been developed utilising symmetric key cryptography, including Advanced Encryption Standard (AES) [53] [54] [55], Data Encryption Standard (DES) [53] [56], Triple Data Encryption Standard (3DES) [53] and blowfish [53].

In 2001, National Institute of Standard and Technology (NIST) replaced previous encryption standards with AES due to its efficiency, implementation and flexibility [54] [55] [56]. AES is based on the principle of substitution-permutation network [54] whereas DES uses Feistel network to encrypt data [54] [56]. Therefore, AES is faster in both hardware and software in comparison to DES [54] [55]. AES is a block cipher which utilises different key sizes: 128-bits, 192-bits and 256-bits [54] [55] [56]. As such, AES also outperforms DES regarding

security, thanks to the large key sizes [54] [55] [56]. AES has been utilised on various platforms, such as AVR Microcontrollers and NVIDIA graphics cards [54]. Other applications include Automatic Teller Machines (ATMs), RFID tags, Bluetooth, Smart cards and Zigbee systems [54].

AES is a subset of a larger encryption algorithm known as Rijndael [54] [55] [56] [57] [58]. This algorithm was designed by the two Belgian cryptographers Daemen and Rijmen [54] [55] [56] [57]. AES utilises a fixed block size of 128 bits and a key size of 128, 192 and 256 bits, which result in 10, 12 and 14 rounds of operation, respectively [54] [55] [56] [58]. Each round consists of 4 stages: Substitute-Bytes, ShiftRows, MixColumns and AddRoundKey [54] [55] [59]. The initial round comprises a 128-bit exclusive or (XOR) of the plaintext with the key to form a new 128-bit state [54] [55] [58] [59]. Each middle round performs the previously mentioned 4 stages [54] [55] [59]. Finally, the last round is identical to each middle round with the exception that the MixColumns stage is not utilised [54] [55]. The decryption process is a reversed version of the encryption rounds [54] [55]. AES has proven to be a well-rounded and reliable encryption method, where the only vulnerability to attacks has been due to attacks exploiting side-channel weaknesses [54] [55]. Side-channel attacks exploit vulnerabilities in the way ciphers are implemented, rather than a weakness within the cipher itself [54] [55].

DES was developed by IBM, based on the Lucifer cipher, back in 1972 [58] [60] [61]. Like AES, DES is a block cipher, applying cryptographic key and algorithms to blocks of data simultaneously, rather than one bit at a time [60] [61]. Each block is enciphered utilising a secret key into a 64-bit ciphertext [60] [61]. DES process involves 16 rounds and can operate in four distinctive modes [60] [61]. In comparison to AES, DES block size is far smaller and requires additional rounds to complete the encryption process [54] [56] [60] [61]. It is for this reason that AES outperforms DES in terms of hardware and software [54] [56]. 8-bits of the 64-bit key are reserved for parity checks, thereby limiting DES key to 56-bits only [60]. This means that a brute force attack would require a maximum of 2^{56} attempts to break [60] [61]. As DES was the first standard symmetric encryption algorithm, it laid the foundation for the field of cryptography [60] [61].

In mid-1990, it was discovered that DES was susceptible to exhaustive key search attacks [60] [61]. Prior to this, DES had shown a robust resistance to known analytical attacks such as differential attack, cipher-text-only and statistical attacks, as well as brute-force attacks [60]. To overcome this problem, a solution was identified to run the DES process three times, creating the 3DES algorithm [60] [61]. This solution added further defence against known attacks, especially brute force which would now have a complexity of 2^{112} [60] [61]. 3DES would achieve higher security levels at the cost of additional problems, namely, that this method required triple resources and latency [60] [61]. As such this process was far slower than standard DES and, therefore, not suitable for real-time applications, as well as being unable to operate on small devices [60] [61].

Blowfish is another form of symmetric encryption which has shown to perform better than DES, as well as possessing high levels of security and speed [58] [62] [63]. This level of security is due to strong key size which enables it to be used in many applications, such as Bulk Encryption, Random Bit Generation and Internet Based Security [58] [62] [63]. In comparison to other encryption techniques, such as DES, 3DES and AES, Blowfish has been shown to possess the best performance regarding the amount of computing resources used [62] [63].

These resources include CPU time, memory and energy consumption [58] [62] [63]. Blowfish has also shown high performance speeds in comparison to other encryption algorithms [62]. This algorithm was created by Bruce Schneier in 1993 as a free and fast symmetric encryption method [58] [62] [63]. The encryption key utilised by blowfish varies between 32-bits and 448-bits [58] [62] [63].

Asymmetric key cryptosystems are the second method of cryptography alongside symmetric key cryptosystems [13] [52] [53] [64]. This method uses one keys for encryption of data and a separate key for decryption [53] [58] [64]. These keys are known as public and private keys and are mathematically related [53] [58] [64]. This means that if a public key is used for encryption purposes, the corresponding private key is required for decryption [53] [58] [64]. The same principle is true in reverse, i.e. public key is required for decryption is the private key was used for encryption [53] [58] [64]. Public key encryption is based upon mathematical functions and requires more computations [53] [58]. As such, this method of encryption is poorly suited for small mobile devices [58]. It is noted that symmetric encryption processes are far quicker than asymmetric encryption, and this difference increases linearly as more data requires encryption [53] [58]. However, asymmetric encryption provides greater security and authentication in comparison to symmetric designs [53]. Only the source device which generates the key pair possesses the private key and public keys can be freely distributed to devices that desire encrypted communication to the source device [53] [58] [64]. It is not possible to identify the private key utilising any of the public keys distributed, increasing encryption security [53] [58].

There are multiple encryption methods that utilise asymmetric concepts, such as Rivest-Shamir-Aldeman (RSA), Diffie-Hellman, Elliptic-Curve Cryptography (ECC) and Digital Signature Algorithm (DSA) [53] [58] [64]. RSA algorithm is most widely used for public key encryption, created by Ron Rivest, Adi Shamir and Leonard Aldeman in 1977 [65] [66]. Despite being classified until this date, the concept was originally thought of way back in 1977 [65] [66]. This method is based on a concept that certain mathematical operations are easier to perform in one direction but require additional information to reverse the process [65] [66]. RSA uses the idea that it is relatively easy to multiply, but more difficult to factor, i.e. multiplication can be computed in polynomial time, but factoring time can grow exponentially in relation to the size of the number [65] [66]. As such, RSA security relies on the concept that factoring large numbers is infeasible due to the time required, even utilising supercomputers [65] [66].

Diffie-Hellman algorithm is a method of generating secret keys in a manner where outside parties cannot identify this key by observing the communication channel [67] [68]. This method relies on the difficulty of computing discrete logarithms to ensure security [67] [68]. The concept of Diffie-Hellman is that two parties begin by generating keys to be used to facilitate encrypted communication [67] [68].

ECC cryptography is based on elliptic curve theory, which is utilised to create faster, smaller and more efficient cryptographic keys [69] [70]. As such, ECC generates keys through the elliptic curve equation rather than generation of keys utilising large prime numbers [69] [70]. Before ECC became popular, most public-key algorithms were based on RSA, DSA and Diffie-Hellman [69] [70]. ECC possesses high security levels, capable of providing security levels equivalent to other encryption algorithms while utilising shorter keys [69] [70] [71]. This trait,

along with smaller bandwidth requirements, less power consumption and faster implementation make this a popular and well-rounded encryption method [71]. Examples where this method is more suitable include smart cards, pc cards, wireless devices and other applications where computational power and space is limited [70] [71].

DSA is composed of the signature algorithm and verification algorithm, which are utilised to create private keys and public keys, respectively [65] [72]. In terms of asymmetric encryption, DSA would utilise private keys to sign a message digest of a document while the public key would be utilised for signature verification [65] [72]. Unlike other encryption methods, DSAs are used for digital signatures rather than data encryption [65] [72]. Despite this, DSA is still considered to provide high levels of security and is one of the more popular algorithms alongside RSA [65] [72]. The authentication process involved in DSA is more complicated in comparison to other encryption methods, such as RSA [65] [72].

While conventional cryptography works well on computer systems, they struggle in an internet of things (IoT) environment [73] [74] [75]. As such, lightweight cryptography methods have been proposed in order to circumvent this limitation [73] [74] [75]. Conventional means are ill suited due to requiring too much processing power, physical space and consume too much battery power [73] [74] [75]. Therefore, lightweight cryptography algorithms are tailored for implementation in IoT environments, including sensor network devices, RFID tags and contactless smart cards [73] [74] [75]. The main limiting factor with lightweight cryptography are power requirements, gate equivalents and timing [73]. In lightweight cryptanalysis, the size of keys, blocks and tags are usually considered for multi-key attacks, power of precomputation and brute force attacks [73].

2.1.3 Cryptanalysis

Cryptanalysis is the study of ciphertext, ciphers and cryptosystems with the goal of understanding their function and how to overcome them [76] [77]. As such, this field of study aims to decrypt ciphertexts without knowledge of the plaintext source, encryption key or the algorithm used to encrypt it [76] [77] [78]. While the goal of cryptanalysis is to overcome cryptographic algorithms, results found via this field of study are also utilised by cryptographers in order to improve upon and strengthen or replace existing flaws [76] [77] [78]. Both cryptanalysis and cryptography are aspects of cryptology, the mathematical study of codes, ciphers and algorithms [76] [77] [78].

Researchers within the cryptanalysis field may find methods of attack that completely break an encryption algorithm, meaning retrieval of the original plaintext becomes a trivial matter [76] [77] [78]. However, more common results are the discovery of weaknesses within the design or implementation of cryptography algorithms [76] [77]. As such, the cryptology field can be described as cryptographers constantly attempting to improve and secure information and cryptanalysts attempting to break cryptosystems [76] [77].

Side-channel cryptanalysis is one method of performing decryption of cryptography algorithms [78]. This method utilises weaknesses of implemented characteristics in order to recover secret parameters involved in the computation [78]. As noted previously, side-channel attacks exploit vulnerabilities in the way ciphers are implemented, rather than a weakness within the cipher itself [54] [55]. Due to this, side-channel cryptanalysis is considered very potent in comparison

to standard cryptanalysis [78] [79]. This avenue of attack is also taken into consideration by implementers of cryptographic devices, as this is a known vulnerability in most cryptography algorithms [78] [79].

Side-channel attacks are very dangerous as they compromise the confidentiality of a system [79] [80]. These attacks can also be utilised to perform further attacks, such as brute-force after the side-channel attack has acquired part of a secret key [79] [80]. The ability to recover even small parts of a secret key provides the brute-force attack with a greater possibility of success [79] [80]. As such, this attack is an effective method of obtaining secret information from cryptographic systems [80].

There are many other types of cryptanalysis attacks and techniques which vary depending on available information that cryptanalyst possesses [76] [78] [80]. These attacks include ciphertext-only attack [81] [82], known plaintext attack [81] [83], chosen plaintext attack [81] [84], differential cryptanalysis [85], integral cryptanalysis [86], dictionary attack [87] and the previously mentioned side-channel attack [80].

Ciphertext-only attack (COA) is performed when an attacker only has access to encrypted messages, but knows nothing about the original plaintext data, the encryption algorithm or the cryptographic key used [81] [82]. This attack method requires the least amount of information to crack an encrypted system and reveal any flaws [81] [82]. However, this attack is also the most difficult to perform for an attacker as there is very little information to work with [81] [82].

During the known plaintext attack (KPA), the cryptanalyst has access to some, or all, of the plaintext from the ciphertext [81] [83]. As such, the goal of this attack is to discover the key used to encrypt the message and, therefore, decrypt the message [81] [83]. Once the key is obtained, the remainder of the ciphertext can be decrypted [81] [83]. KPA relies on the attacker being able to discover, or guess, part of an encrypted message or the entire message [81] [83]. One example of obtaining part of the information could be knowing who a message is addressed to [81] [83]. Therefore, utilising this information, part of the original plaintext can be identified [81] [83]. Linear cryptanalysis is a form of KPA [88]. Both linear and differential cryptanalysis are widely accepted by the cryptanalyst community and are standard techniques to be used to analyse existing and proposed ciphers [88]. However, linear cryptanalysis has not had as much success in comparison to differential cryptanalysis [88].

Chosen plaintext attack (CPA) occurs when the analyst either knows the encryption algorithm or has access to the device used to encrypt the data [81] [84]. An analyst is then capable of encrypting the chosen plaintext with the targeted algorithm to acquire information regarding the encryption key [81] [84]. The keys are recovered using selected pairs of plaintexts and ciphertexts [81] [84]. CPA is relatively easy to accomplish, especially in comparison to COA and KPA [81] [84]. The assumption that an attacker is capable of freely selecting the plaintext may not be completely accurate in practical situations [81].

Differential cryptanalysis is a common approach to analysing iterative block ciphers [85] [89]. The goal of this method is to compare the differences in input in relation to the differences in encrypted output when data is exposed to various algorithms [85] [89]. This method is primarily used to determine if changes in plaintext result in any non-random results in the encrypted ciphertext [85] [89]. Should changes prove to be non-random, then it is possible that

the encryption algorithm possesses vulnerabilities [85] [89]. As such, it is possible to obtain information about what was encrypted, or how it was encrypted, by analysing how encrypted data changes from these inputs [85] [89].

Integral cryptanalysis is like differential cryptanalysis, except for utilising sets of plaintexts rather than pairs [86] [90]. These methods are utilised to attack block ciphers whose crypt module is divided into two equal sections [90]. Integral cryptanalysis has been utilised on multiple ciphers, including Rijndael, Skipjack, Camellia, Misty etc. [90]. It was noted that compared with differential and linear cryptanalysis, integral analysis is more effective in many decrypting progresses [90]. Examples include being able to attack more turns or the complexity of attacking is decreased when attacking the same number of turns [90].

Dictionary attacks are typically used against password files [87]. These attacks rely on the human tendency to use passwords based on dictionary words or easily guessed sequences of numbers and letters [87]. Therefore, this attack operates by encrypting all words in the dictionary and then comparing if the resulting ciphertext words match encrypted passwords stored within password files [87].

There are multiple other tools utilised by cryptanalysts in order to obtain certain information, such as network sniffers and password cracking software. Other web-based tools are also available and widely used in order to break cryptosystems, teach about network security and test cryptosystem security levels [91]. These tools include:

- CrypTool [91]. This is an e-learning tool developed in C++ language and launched in 1998 with the purpose explaining the workings of cryptanalysis and cryptography [91]. It aims at helping people understand network security threats and the workings of cryptology [91]. CrypTool includes asymmetric ciphers such as RSA and Elliptic Curve cryptography [91]. CrypTool2 (CT2) was launched in 2014 and possesses several cryptanalysis and cryptography functions [91]. Both CrypTool1 (CT1) and CT2 operate on windows [91].
- EverCrack [91]. This is an open source software which deals primarily with mono-alphabetic substitution and transposition ciphers [91]. It has multilanguage support and was developed in C language [91]. The overall goal is to break down complex ciphers systematically into their simplex components for cryptanalysis [91]. Currently, the main concentration is on online web-based applications [91].
- Cryptol [91]. This software was developed by Galois Inc. as a learning tool and a method to analysing algorithms and implementations [91]. Cryptol is designed in a way that allows cryptographers to watch how stream processing functions to manipulate the ciphers or encryption algorithms [91]. This software deals with cryptographic routines as opposed to entire cryptographic suites [91]. Cryptol was designed for efficient implementation of cryptographic algorithms into hardware circuits [91]. Despite this software's flaws, it can improve the trustworthiness of many cryptographic algorithms [91].
- AlphaPeeler, which is a non-commercial software product for educational and personal use [91]. This software includes ciphers such as MD5, SHA-1, RSA and classic ciphers such as Playfair [91]. The main purpose of this software is to provide up-to-date research to assist businesses, vendors and carriers [91].

- Crypto Bench, a software capable of performing various cryptanalytic functions [91]. This software can generate 14 cryptographic hashes and two checksums [91]. Crypto Bench can encrypt with 29 different secret keys or symmetric encryption algorithms and it is able to encrypt, decrypt, sign and verify with six different public key or asymmetric algorithms [91].
- Ganzúa is a Java application capable of running on Mac, Linux or Windows and is a cryptanalysis tool for monoalphabetic and polyalphabetic ciphers [91]. This software has the capability of operating with almost any set of characters [91]. Ganzúa possesses features to cryptanalyze both monoalphabetic and polyalphabetic ciphers [91]. These include alphabet wide substitution tools for Caesar shift cipher and relative frequencies of characters, bigrams and trigrams [91].

2.1.4 Distributed Network Attacks (DNA)

Distributed Network Attacks (DNA) is an approach to recovering password protected files [92] [93] [94]. Typically, recoveries are limited to the processing power of one machine, but with DNA, the power of multiple machines across the network or across the world can be used to decrypt passwords [92] [93] [94]. Devices running DNA software and participating in this operation are known as client machines [92] [93] [94]. These machines must be capable of accessing a central DNA server over the network [92] [93] [94]. A DNA Manager then coordinates the attack by assigning small portions of the key search to machines distributed across the network [92] [93] [94]. This attack is performed in the background of client devices, and these DNA clients only utilise unused processor power and time to perform this attack [92] [93] [94].

DNA was designed by AccessData with the intent of performing distributed decryption attacks across large enterprises [92] [95]. These attacks are carried out using grid implementation [92]. The cost of maintaining the grid side of the product was high, as client applications required regular updates to support several exchanging operating systems proved time consuming [93]. Furthermore, debugging distributed communications began to consume resources that could be used more efficiently in other areas [93].

In order to alleviate issues encountered with DNA, and to improve upon the software itself, a collaboration between Parabon and AccessData enabled the creation of DNA with Frontier [96]. This was done by integrating DNA into Parabon's Frontier grid platform [96]. Parabon would go on to improve their Frontier grid interface to allow DNA to perform tasks using its native format, greatly aiding within the integration process [96]. The result of this collaboration is a product that possesses both the decryption capabilities of DNA, along with the reliability and scalability of Frontier [96]. DNA with Frontier is capable of decryption that can scale across thousands of computers [92] [96]. This software is said to run smoothly, as if only utilising a single host, while possessing high decryption speeds [94] [95].

2.2 Related Work

While general literature review into various journals, books and other online sources has provided large amounts of information on cryptography and cryptanalysis, a more focused analysis of related work can build upon this knowledge, allowing for more precise tests to be

constructed. One paper, written by S. J. Aboud, looked at RSA encryption and decryption methodologies [97]. More specifically, the paper looks at possible alternative attacks to RSA in order to break this encryption scheme [97]. RSA encryption schemes are based on the difficulty to factor large integer numbers for security purposes [97]. Factoring an integer means finding its prime numbers [97]. In RSAs case, this means discovering p and q values from the public keys N value [97].

S. J. Aboud discussed another method of RSA attack involving a small decryption exponent d , where the d value is at least 160 bits to avoid square root discrete logarithms [97]. He noted that this attack, however, can be avoided if the encryption exponent e is at least 50% longer than modulus N [97]. As such, Aboud [97] proposed an algorithm to break without the requirement of factoring N [97]. Factoring of N is typically an obstacle to overcome in cryptanalysis, in order to reconstruct an RSA private keys for decryption purposes [65] [66] [97]. Aboud's [97] proposed algorithm was designed to prove a possibility that if the exponent e is known, then N can be factored in polynomial time by means of randomised algorithm [97].

This paper presented a scenario where the values of a public key are set as 8927 for N and 2621 for e [97]. Furthermore, values of the private are set to 5 for exponent d , 113 for p and 79 for q [97]. From here, an entity titled B, would encrypt a message (m) so the cipher text (c) is equal to 6948, then this message would be transmitted to entity A [97]. However, entity T has intercepted the message and will attempt to recover the original content [97]. A series of mathematical equations are then run, which can be seen on page 589 of the report [97].

It is stated that they had run this scheme 100 times in order to find an average key generation time, which was noted at 93 ms in comparison to 1342 seconds of RSA-small- e scheme [97]. The time difference is due to RSA-small- e schemes factoring N in order to isolate p and q values, whereas in the scheme proposed by Aboud, these values were not computed [97]. Aboud's desired goal was to isolate small factors until specific factoring was found [97]. They had demonstrated a more efficient scheme in comparison to RSA-small- e schemes [97]. However, it should be noted that this attack, and subsequently the RSA-small- e scheme, are designed with a singular target in mind. In this specific scenario, these schemes are based on the concept of a small private key d value.

As such, while this attack scheme is efficient in isolating desired prime numbers through factoring, the target for this specific attack is small. Typically, RSA keys security is based on large integers where factoring these N values becomes a monumental task. Similarly, typical RSA encryption utilises an e value of 65537 [65] [66]. As such, d values for private keys are typically large integers that this scheme would struggle in properly factoring. Therefore, this scheme is efficient for its intended purpose, but would become less so as RSA key sizes increase.

Another paper, written by N. Floissac and Y. L'Hyver [98] was examined as part of related work. This paper analyses AES encryption, with a focus primarily on 192- and 256-bit AES [98]. They discuss a specific attack, known as Differential Fault Analysis (DFA), which allows the discovery of secret keys by corrupting intermediate states in order to produce faulty ciphers [98]. The attack is based on analysing the differences between the faulty ciphers and the expected ones to obtain information on the secret key [98]. This attack has been widely used against AES-128 with a shift in focus towards 192- and 256-AES keys [98].

Floissac and L’Hyver propose a form of this attack to be used against AES-192 and -256 by exploiting faults on KeyExpansion computation [98]. The attack is based on the AES-128 counterpart, but due to differences in KeyExpansion algorithms between different AES key sizes they were required to analyse the specific fault impacts within the key to overcome these problems [98]. One problem they encountered was overcome by altering the equation so that it takes into consideration specific vault values during one round or the AES process [98]. As such, they were able to identify the K_{nr} key value and the last four bytes of the K_{nr-1} [98]. Another problem encountered was with AES-256, as it required an additional 12 bytes to be discovered through the reproduction phase [98].

Through their testing, Floissac and L’Hyver identified two main adaptations to allow DFA to be utilised against higher AES key sizes [98]. Firstly, they extended the original attack, which allowed them to gain access to K_{nr} key values and the four bytes of the K_{nr-1} key, as previously mentioned [98]. Secondly, a reproduction of the attack could be utilised to identify the third column of K_{nr-1} [98]. They did propose an alternative as much of K_{nr-1} had been identified and, therefore, an exhaustive search could be utilised to identify the remainder [98]. They concluded that at least eight couples are enough in retrieving the entire missing column and finally the initial AES key [98].

Floissac and L’Hyver presented an avenue in which higher AES keys can be retrieved [98]. However, for the scope of this project, their method of attack far exceeds the authors knowledge. As such, this paper was used as a reminder that there are many possible methods to use in order to acquire AES security keys. Not only this, but it improves upon the authors knowledge of AES functions by examining AES rounds of encryption, and how additional rounds added in AES-192 and AES-256 impact various attacks, e.g. attacks that can work against AES-128 may have less success against AES-192 and AES-256. It also helped in eliminating one potential attack used as the setup far exceeds the limits of this project.

Chapter 3

3.1 Methodology

The literature review proved instrumental in identifying key aspects of the project. This project is to compare the performance of DNA in relation to other decryption software. Therefore, research into different encryption methods and algorithms would aid in understanding their function. Furthermore, research into the field of cryptanalysis provided data on various decryption methods and software. The software's discovered from research can then be used to encrypt and decrypt data. Following this, the performance of decryption methods can be compared to DNA software. This process can be used for multiple encryption methods identified via literature review.

3.1.1 Philosophies

The intention of this project is to analyse the efficiency of DNA software in comparison to other decryption methods. As such, this project takes a Positivism approach to data methodology. In positivism studies the role of a researcher is limited to data collection and interpretation. Positivism research refers to researchers which believe only data acquired via observation, including measurement, is trustworthy. As this project aims to acquire data via comparison of multiple software, and therefore using the results to come to a specific conclusion, this project meets the criteria for positivism research.

This project philosophy can also be viewed in a Pragmatism manner. Pragmatism research recognises that there are many ways of undertaking research, and that no single point of view can ever give the entire picture. The field of cryptology can effectively be split into both cryptography and cryptanalysis. As such, research into both fields was required in order to gain an understanding of the overall field of study. Pragmatists are not required to use multiple methods to obtain information. Rather, they utilise a combination of methods that advances a specific research in an efficient manner.

3.1.2 Approach

The aim of this project leads more towards a deductive approach to research. Deductive approaches are based on developing a hypothesis based on an existing theory, then designing a strategy in order to test this hypothesis. In other words, a deductive approach is focused on obtaining and analysing a conclusion from a hypothesis.

In the case of this project, the aim is to compare DNA software to other cryptanalysis software in order to determine its efficiency in decrypting ciphertext. As such, the hypothesis is that DNA would be capable of decrypting ciphertext faster as it utilises unused processing power of multiple machines to accomplish tasks. In relation to a deductive approach, this project hypothesis will be tested by running a series of tests in order to prove or disprove this hypothesis. Therefore, this project will deduct conclusions based on the previous mentioned hypothesis and corresponding strategy.

3.1.3 Strategies

There were many strategies utilised in the completion of the project, with research and experimentation being the two prominent methods. The main research strategy could be described as case study or archival research. Case studies involve study into a specific idea, or hypothesis, and using multiple sources of evidence to back up findings. Archival research is based on administrative records and documents as sources of data. For this project, the goal of research was to gain greater understanding into fields associated with the project topic. In this case, research into cryptography and cryptanalysis was required to gain knowledge in these fields, which aided in the construction of an experiment in which to prove or disprove the project hypothesis.

Research was conducted using online journals found on sites such as IEEE Xplore and Science Direct. Other sources included library books based on cryptography and cryptanalysis. These sources were chosen due to being peer-reviewed by multiple individuals and groups in the same field of study. As such, if the sources are up to date, the information within can be considered accurate. The research into these fields identified several encryption methods which are widely utilised in modern networks. These included symmetric and asymmetric encryption methods [13] [52] [53]. From here, several encryption algorithms were identified, such as AES [54] [55], DES [58] [60] and blowfish [62] for symmetric encryption, and RSA [65], Diffie-Hellman [67] and ECC [69].

Other critical information acquired via research was into cryptanalysis. This information included what is cryptanalysis, what attacks are performed and what tools can be used for cryptography and cryptanalysis purposes. Certain tools such as CrypTool, Cryptol and Crypt Bench were identified as possibilities for this project [91]. Finally, research into DNA was done in order to gain greater understanding of this software's function [92] [93] [94]. One point of note was the lack of peer-review journals which utilised this software in the manner proposed within this project. As such, related work including the use of DNA could not be identified. This led to a lot of experimental research into the effectiveness of DNA in order to gain an understanding of its functions.

This leads into the final research strategy utilised, i.e. experimentation. As mentioned, the aim of this project is to determine whether DNA was more effective than standard decryption methods and software. Due to the nature of this project, experimentation with all software utilised needs to be conducted to gain an understanding of their purposes and limits. Finally, encrypted data would be generated, and subsequently decrypted utilising various software mentioned, i.e. CrypTool, DNA etc.

Once all experimentation has been concluded, data would be gathered and compared with results concluded from each test. The main bits of information which would be used include if DNA was successful in decrypting ciphertext in comparison to another cryptanalysis software. Furthermore, from these results it can be hypothesised if utilising DNA in this manner is truly efficient or not.

3.1.4 Choices

The project will adopt a mono method form of data analysis. This is due to the nature of data acquired and analysed, i.e. it takes a more quantitative form. In quantitative study, the data is

in numerical form and is analysed using quantitative analysis techniques. Quantitative research utilises measurable data to formulate facts and discover patterns in research. In relation to this project, the data will be obtained from observation of experiments conducted. This data will include time taken to decrypt various ciphertexts and whether certain software was successful in decrypting specific ciphertexts. These results can then be compared with similar experiments in order to determine if the results found match results discovered from other experiments.

3.1.5 Time Horizons

This project will take a more cross-sectional approach to time frame observations. Cross-sectional refers to a one point in time observation of data as well as utilising different samples of data. The nature of this project will lead to the utilisation of different software, encryption ciphers and ciphertext. As the data will be presented once the experiment has concluded, this makes the project a one point in time type of observation of data. There is, however, one exception to this rule. If a software is shown to be unable to complete the decryption process in a realistic timeframe, the data will not be obtained at the end and instead be obtained at that specific point in time. This reinforces the concept of a cross-sectional time frame, as the data will still only be acquired at one point in time.

3.1.6 Techniques and Procedures

Once all experiments have been concluded, the data collected will determine whether the project aim was successful or not. In this scenario, was DNA more effective at decryption processes in comparison to another cryptanalysis software. The collected data will be compared to expected results and data from similar experiments. Expected data will be obtained from the literature review research, i.e. expected outcome of success based on understanding of various encryption ciphers.

3.1.7 Risk Assessment

As with many projects, there are risks involved in both setup and completion. This refers to the fact that there are many possibilities and outcomes that could arise that are unpredictable. Therefore, precautions must be set to ensure that the project is not disrupted in the event of potential issues that could arise. Due to the usage of multiple types of software, one critical risk identified is the inability to obtain specific software, or perhaps the software does not function as expected. As such, it is critical that software used within this project can be readily obtained without too much trouble.

Another risk involved with the project is data. All projects utilise data in one form or another, therefore, all projects are vulnerable to these risks. These risks include the possibility of data becoming corrupted, lost or damaged. As such, all data obtained from experimentation must be noted clearly and subsequently backed up to ensure it is not lost. Copies of all data obtained should not be stored within the same area, thus limiting potential damage caused by any unforeseen events.

Chapter 4

4.1 Design

This goal of this project is to test the effectiveness of various cryptographic methods, such as AES, DES and RSA. An analysis of their strengths and weaknesses will be examined utilising various software. Firstly, this project utilised a comprehensive literature review in order to determine the best design to accomplish the aim set within the project. The three main cryptographic algorithms chosen were AES, DES and RSA. These were chosen as they are widely used in today's encrypted communications for various services and businesses. This choice also presents a balance of algorithms, i.e. AES and DES represent symmetric encryption methods while RSA represents an asymmetric encryption method. As such, the analysis of this project encompasses both common methods of data encryption, allowing for a comparison of both methods to determine their effectiveness and suitability for deployment in real world scenarios.

One software which will be utilised is Distributed Network Attack (DNA), created by AccessData [92] [93] [94]. AccessData is a forensics company focusing on various software for cryptography, cryptanalysis and other forensics needs [95]. Other known software created by AccessData includes the Forensics Toolkit (FTK), a forensics software suite which is popular among digital forensics professionals [95]. As such, AccessData has shown to deploy various software which can be utilised for various investigatory tasks. Such tasks include cryptanalysis work, through their DNA software.

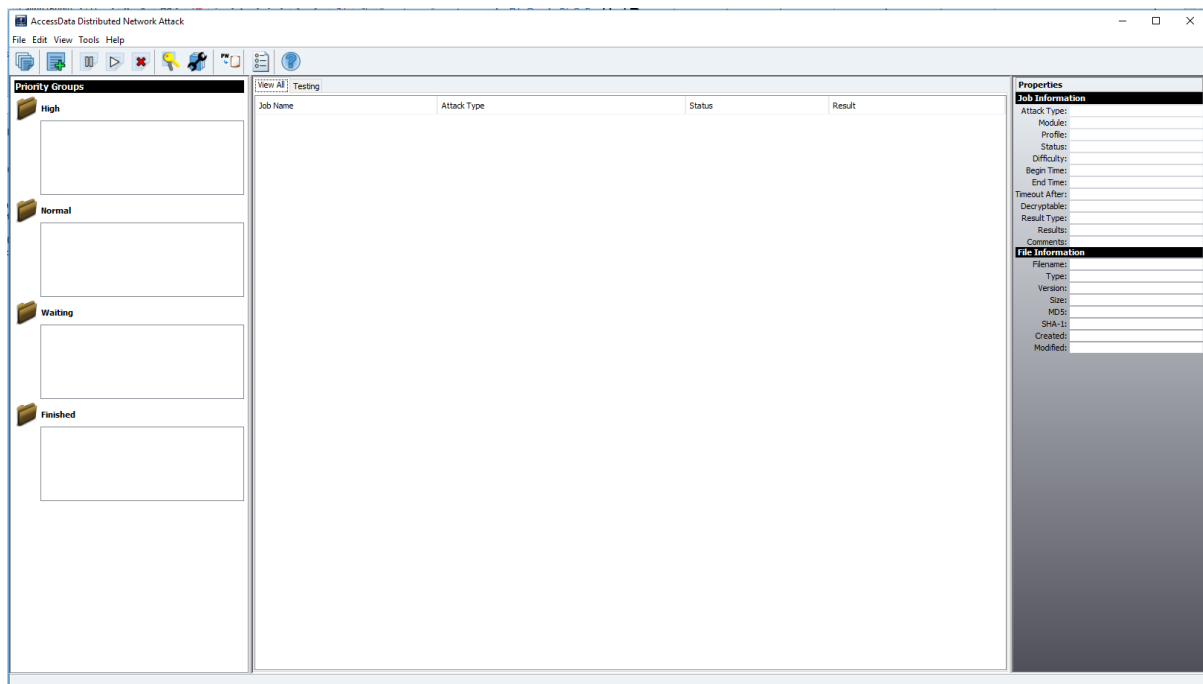


Figure 1: DNA Layout

Figure 1 depicts the DNA software which can be utilised to share cryptanalytic tasks across multiple devices. The result of this is utilising unused processing power within devices to work towards a single task, i.e. decryption. A setup will be created where one device would act as the DNA server and multiple other devices within the same network would be set up as DNA clients. The DNA server is required within a central location and would allocate tasks to DNA clients. These tasks would be allocating small portions of the key search to machines distributed

throughout the network. A DNA manager would coordinate this attack. The goal of this approach is to lessen time required to acquire cryptographic keys used in the encryption / decryption process, thereby making ciphertext data accessible.

As a point of comparison, another software will be used to determine the effectiveness of various decryption methods. This would also demonstrate the effectiveness of DNA in comparison to other software. In theory, DNA should prove superior as it would utilise multiple devices in cryptanalysis attacks where other software would only utilise the processing power of a single machine. CrypTool 2 will provide the perfect setup as it allows for messages to be encrypted and enables multiple methods of cryptanalysis attacks [91] [99]. This software utilises C++ language and is shown to possess several cryptanalysis and cryptography functions [91] [99].

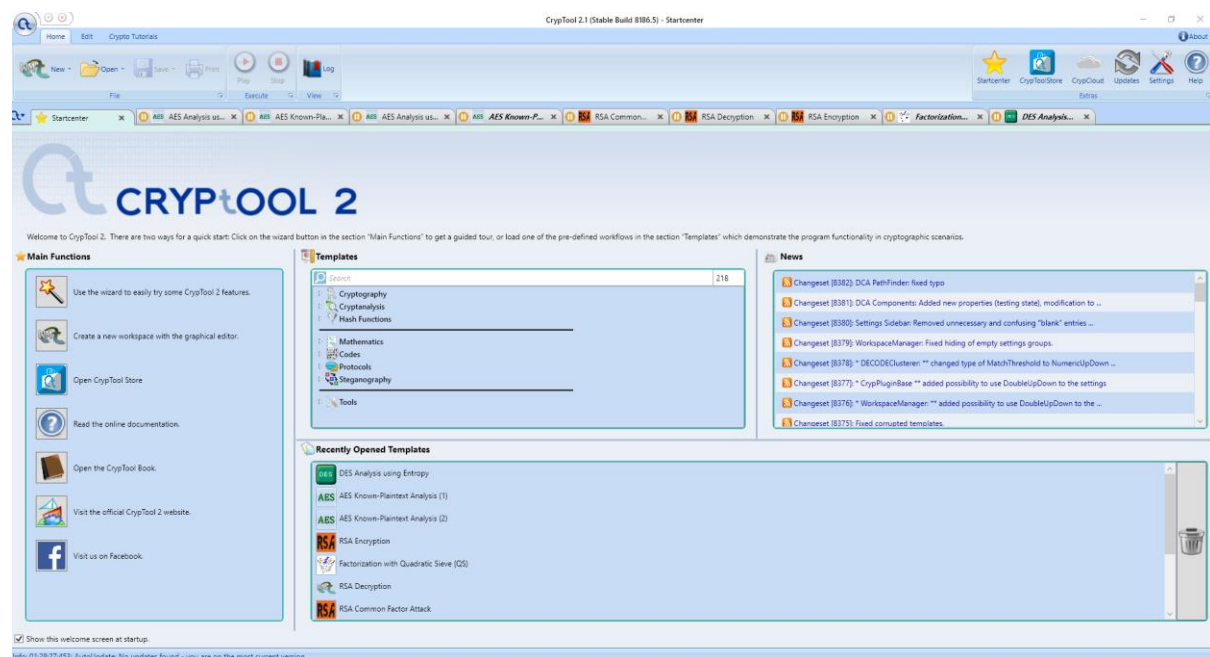


Figure 2: CrypTool 2

Figure 2 shows the CrypTool 2 software screen. As seen, there are numerous functions within this software regarding both cryptography and cryptanalysis. Both classical and modern ciphers are accessible from this software, including AES, DES and RSA. Research found while conducting a literature review showed this software capable of performing Ciphertext-only and known plaintext attacks [81] [82] [83]. This software also clearly shows the process involved in ciphertext attacks, the time required to complete and can be used to determine if these attacks are realistic in a real-world scenario. The process involved in analysing asymmetric key decryption is also available, such as factoring numbers in order to obtain RSA private key information.

Finally, Kali will be utilised to demonstrate the process of various attacks utilised to compromise cryptographic security integrity. Kali is a Debian-based Linux Operating System aimed at advanced penetration testing and security examining [100]. This software possesses several built-in penetration testing tools [100]. For the scope of this project, Kali will be utilised to demonstrate the process of exploiting cryptographic vulnerabilities that exist from their implementation, rather than due to their design. The literature review described one such attack

which was the side-channel attack [79] [80]. This attack can be utilised to acquire parts of a cryptographic secret key, enabling attacks which will be performed utilising CrypTool 2 [79] [80] [91] [99]. Side-channel attacks are very dangerous as they compromise the confidentiality of a system [79] [80]. The ability to recover even small parts of a secret key provides the brute-force attack with a greater possibility of success [79] [80]. As such, this attack is an effective method of obtaining secret information from cryptographic systems [80].



Figure 3: Kali Linux OS

Figure 3 shows the Kali OS from which numerous scanning and exploitation attacks can be conducted for penetration testing purposes [100]. Kali was run utilising Oracle Virtual Machine software, which allows a device to run several virtual machines. For the purposes of this project, Kali will be used to discuss how to perform specific attacks and scans that target vulnerabilities in cryptographic libraries. These vulnerabilities allow a user to obtain information regarding encryption methods used, as well as possibly obtaining the encryption key, ciphertext example or both.

4.2 Implementation

The implementation section will discuss the setup of each attack used against each encryption method. Three main software were utilised throughout the course of this project: Kali using Oracle VM; CrypTool 2; and Distributed Network Attack (DNA) by AccessData. The first section demonstrated potential implementations of kali in order to launch scans to determine vulnerabilities within cryptography implementations. From here potential attacks were shown

to be possible in order to obtain information regarding cipher texts, secret keys etc. However, due to ethical reasons, only scanning was conducted in order to ensure no damage is caused to any site or device. The purpose was to demonstrate how to obtain ciphertext information, such as potential clear text or cipher keys. As such, testing utilising CrypTool and DNA assumes successful targeting of a vulnerable site with further success in conducting vulnerability attacks in order to obtain cryptographic information.

4.2.1 Kali Implementation

The usage of Kali within this project was not extensive, as several attacks go beyond the scope of this task. Kali's purpose was to demonstrate and discuss several cryptographic vulnerabilities that exist outside of the actual cipher design. One command line identified was to discover if a server possessed a Heartbleed vulnerability.

```
root@kali:~#  
root@kali:~# nmap -sV --script=ssl-heartbleed [target]
```

Figure 4: Heartbleed Vulnerability Scan

Using nmap, it is possible to scan a target in order to determine if the system is vulnerable to Heartbleed exploits. HTTPS provides secure communication in comparison to standard HTTP, as data communication utilising SSL should be secure and encrypted. However, it is possible to recover unencrypted information from heartbeat messages [101]. The command shown in Figure 4 can be utilised to determine if a device is vulnerable to heartbeat attacks. If a target is vulnerable to this scan, an output shown in Figure 5 will be displayed. Otherwise, a display shown in Figure 6 would be shown as there are no vulnerabilities detected.

```
Starting Nmap 6.40 ( http://nmap.org ) at 2014-04-20 17:49 EDT  
Nmap scan report for 192.168.1.70  
Host is up (0.00020s latency).  
Not shown: 997 closed ports  
PORT      STATE SERVICE VERSION  
22/tcp    open  ssh      OpenSSH 6.0p1 Debian 4 (protocol 2.0)  
80/tcp    open  http      Apache httpd 2.2.22 ((Debian))  
443/tcp   open  ssl/http  Apache httpd 2.2.22 ((Debian))  
| ssl-heartbleed:  
| VULNERABLE:  
|   The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. It allows for stealing information intended to be protected by SSL/TLS encryption.  
|   State: VULNERABLE  
|   Risk factor: High  
|   Description:  
|     OpenSSL versions 1.0.1 and 1.0.2-beta releases (including 1.0.1f and 1.0.2-beta1) of OpenSSL are affected by the Heartbleed bug. The bug allows for reading memory of systems protected by the vulnerable OpenSSL versions and could allow for disclosure of otherwise encrypted confidential information as well as the encryption keys themselves.  
|   References:  
|     http://www.openssl.org/news/secadv_20140407.txt  
|     https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160  
|     http://cvedetails.com/cve/2014-0160/  
MAC Address: 00:0C:29:37:4A:26 (VMware)  
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Figure 5: Heartbleed Vulnerability Scan Output [101]

```

Starting Nmap 7.80 ( https://nmap.org ) at 2019-09-25 22:06 UTC
Stats: 0:04:32 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 90.91% done; ETC: 22:10 (0:00:07 remaining)
Nmap scan report for uwtzd.ac.uk (212.219.178.84)
Host is up (0.021s latency).
rDNS record for 212.219.178.84: www.tsd.ac.uk
Not shown: 988 filtered ports
PORT      STATE SERVICE      VERSION
25/tcp    open  smtp?
80/tcp    open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
| http-server-header:
|   Microsoft-HTTPAPI/2.0
|   Microsoft-IIS/7.5
110/tcp   open  pop3?
113/tcp   closed ident
119/tcp   open  nntp?
143/tcp   open  imap?
443/tcp   open  ssl/http     Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
| http-server-header:
|   Microsoft-HTTPAPI/2.0
|   Microsoft-IIS/7.5
|_http-trane-info: Problem with XML parsing of /evox/about
465/tcp   open  smtps?
563/tcp   open  snews?
587/tcp   open  submission?
993/tcp   open  imaps?

```

Figure 6: Heartbleed Vulnerability Scan Alternate Output

In an event that a vulnerability exists, another Kali tool can be utilised to perform an attack and extract data from a server's memory. This software is known as Metasploit [100] [101].

```

# cowsay++
< metasploit >
-----
      \      (oo)_____)
       \      (__)_____)
          ||--|| *

      =[ metasploit v5.0.41-dev ]
+ -- --=[ 1914 exploits - 1074 auxiliary - 330 post ]
+ -- --=[ 556 payloads - 45 encoders - 10 nops ]
+ -- --=[ 4 evasion ]

msf5 >

```

Figure 7: Metasploit Within Kali

```
msf5 > use auxiliary/scanner/ssl/openssl_heartbleed
msf5 auxiliary(scanner/ssl/openssl_heartbleed) > show options

Module options (auxiliary/scanner/ssl/openssl_heartbleed):

  Name          Current Setting  Required  Description
  ----          -
  DUMPFILTER     none             no        Pattern to filter leaked memory before storing
  LEAK_COUNT     1               yes       Number of times to leak memory per SCAN or DUMP invocation
  MAX_KEYTRIES   50              yes       Max tries to dump key
  RESPONSE_TIMEOUT 10             yes       Number of seconds to wait for a server response
  RHOSTS         nil              yes       The target address range or CIDR identifier
  RPORT          443             yes       The target port (TCP)
  STATUS_EVERY   5               yes       How many retries until key dump status
  THREADS        1               yes       The number of concurrent threads
  TLS_CALLBACK   None            yes       Protocol to use, "None" to use raw TLS sockets (Accepted: None, SMTP, IMAP, JABBER, POP3, FTP, POSTGRES)
  TLS_VERSION    1.0             yes       TLS/SSL version to use (Accepted: SSLv3, 1.0, 1.1, 1.2)
```

Figure 8: Metasploit Heartbleed Options

Utilising the option seen in Figure 8, it is possible to run an SSL Heartbleed exploit attack. However, due to the limitations of this project, this scan will not be run. Rather, an example shown in Figure 9 shows the results of this very same scan performed during another paper [101]

```
msf auxiliary(openssl_heartbleed) > run

[*] 192.168.1.70:443 - Sending Client Hello...
[*] 192.168.1.70:443 - Sending Heartbeat...
[*] 192.168.1.70:443 - Heartbeat response, checking if there is data leaked...
[+] 192.168.1.70:443 - Heartbeat response with leak
[*] 192.168.1.70:443 - Printable info leaked: SSF@CDpkDRM5k}?`f"!98532ED/A8M`i@K4-cM@5@`K4@pK84@lowercase_octets
es/canonical.php0x7f15876b6b2f5@d*P$Xp6@6@3@5@redirect_canonical`7@Nxp`6HX$`SPp`8@p7@5@H7@_remove_qs_args_if_not
irect_guess_404_permalinkP=@N03$(8`$0p<@P:@p8@(:@wp_redirect_admin_locations@I0@@(Yh#<@3@. M@`0A@:@9@:@add_shor
;@8=@remove_shortcodeX=@>@NB80DxlpHDBp?@>@<@>@remove_all_shortcodeserialized_st`D@NDxP8Fx|PFD pC@@@>@@@shortcode
IXIG`B@A@?@A@has_shortcodetiB@>@SJ8T"S@TxPTJ``@H@C@A@B@do_shortcode`D@>@:U[H[x[
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figure 9: Metasploit Heartbleed Results [101]

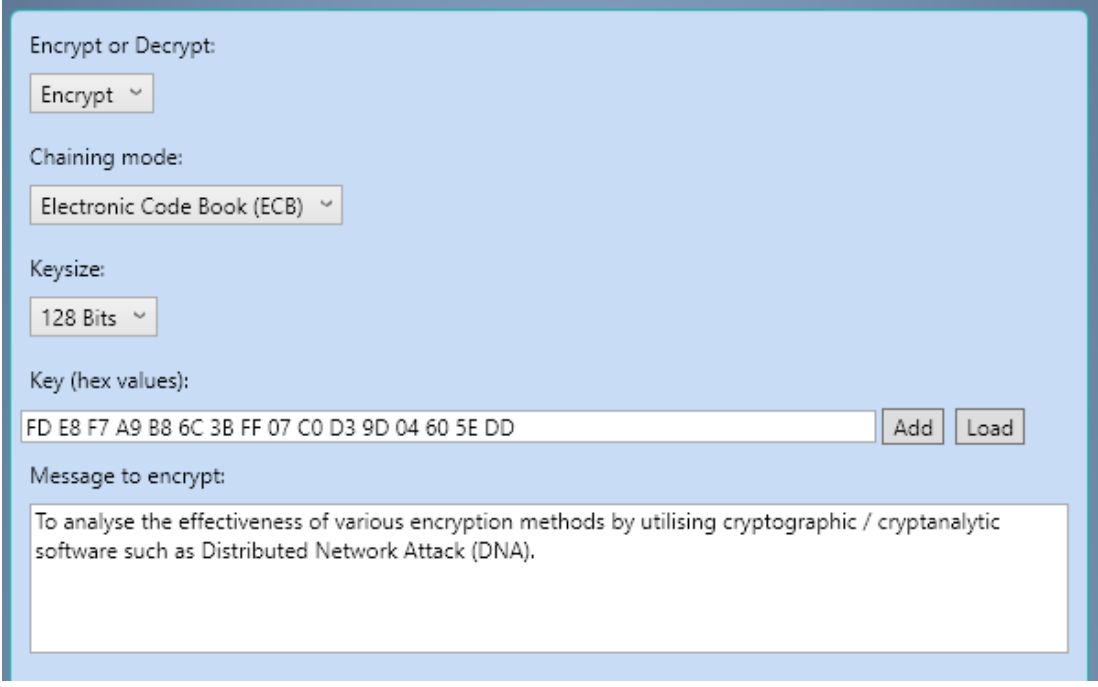
It was noted that random data was successfully pulled from their test server, and if the same scan was run again, additional data could also be pulled [101].

Kali can also be utilised to determine if a target is vulnerable to the Padding Oracle on Downgraded Legacy Encryption (POODLE) or Browser Exploit Against SSL/TLS (BEAST) attacks. By performing port scans utilising nmap, it is possible to identify cipher suits utilised on this port. From these scans, example seen once again in Figure 6, if SSLv3 ciphers were identified, BEAST attacks could be used to exploit weaknesses in the cipher block chaining (CBC). This would enable Man-in-the-Middle (MitM) attacks, which would allow an attacker to decrypt and obtain authentication tokens, giving access to data transmitted.

4.2.2 CrypTool 2 Implementation

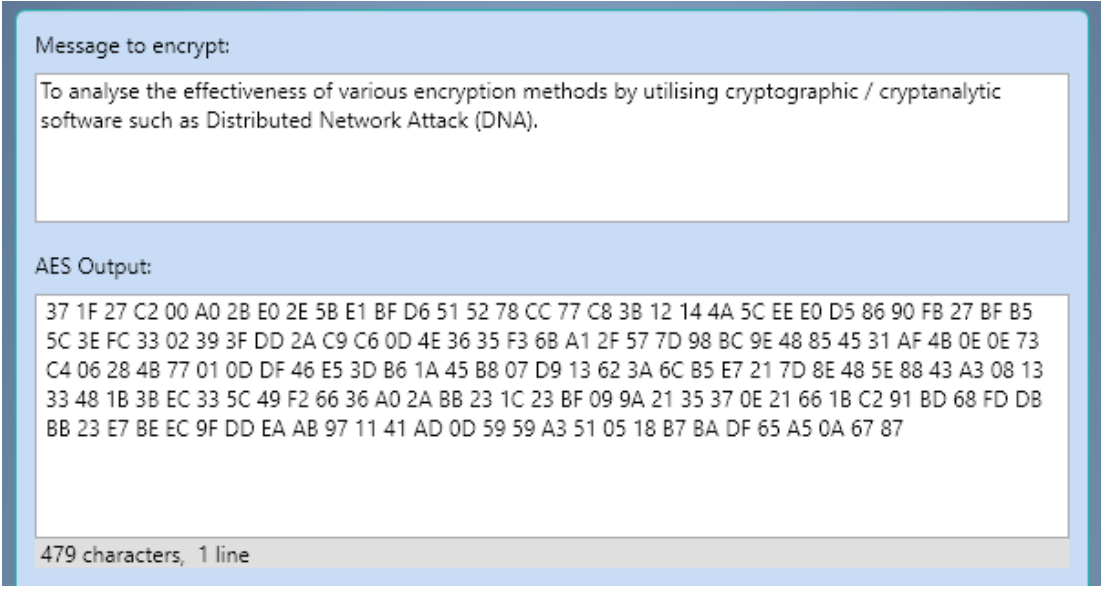
As stated previously, CrypTool was chosen due to it possessing several encryption and decryption features. In order to properly test various decryption methods, it became necessary to acquire encrypted data. Kali techniques were discussed as methods of acquiring cryptographic information and the following tests assume a successful acquisition of information. This allows for testing of decryption method in order to determine the strength and reliability of various encryption algorithms. These results can then be analysed alongside findings discovered from related work to determine key effectiveness in real world scenarios.

The first encryption algorithm tested was AES. Figures 10 and 11 show the encryption process involved to generate ciphertext to be used in decryption tests.



The screenshot shows the CrypTool 2 encryption interface. It has a light blue background. At the top, there's a section titled "Encrypt or Decrypt:" with a dropdown menu set to "Encrypt". Below that is "Chaining mode:" with a dropdown menu set to "Electronic Code Book (ECB)". Then "Keysize:" with a dropdown menu set to "128 Bits". Below that is "Key (hex values):" with a text input field containing "FD E8 F7 A9 B8 6C 3B FF 07 C0 D3 9D 04 60 5E DD" and two buttons labeled "Add" and "Load". At the bottom is "Message to encrypt:" with a text area containing the text: "To analyse the effectiveness of various encryption methods by utilising cryptographic / cryptanalytic software such as Distributed Network Attack (DNA)."

Figure 10: AES Encryption Process



The screenshot shows the output of the AES encryption process. It has a light blue background. At the top, there's a section titled "Message to encrypt:" with a text area containing the text: "To analyse the effectiveness of various encryption methods by utilising cryptographic / cryptanalytic software such as Distributed Network Attack (DNA).". Below that is "AES Output:" with a text area containing the ciphertext: "37 1F 27 C2 00 A0 2B E0 2E 5B E1 BF D6 51 52 78 CC 77 C8 3B 12 14 4A 5C EE E0 D5 86 90 FB 27 BF B5 5C 3E FC 33 02 39 3F DD 2A C9 C6 0D 4E 36 35 F3 6B A1 2F 57 7D 98 BC 9E 48 85 45 31 AF 4B 0E 0E 73 C4 06 28 4B 77 01 0D DF 46 E5 3D B6 1A 45 B8 07 D9 13 62 3A 6C B5 E7 21 7D 8E 48 5E 88 43 A3 08 13 33 48 1B 3B EC 33 5C 49 F2 66 36 A0 2A BB 23 1C 23 BF 09 9A 21 35 37 0E 21 66 1B C2 91 BD 68 FD DB BB 23 E7 BE EC 9F DD EA AB 97 11 41 AD 0D 59 59 A3 51 05 18 B7 BA DF 65 A5 0A 67 87". At the bottom, there's a status bar that says "479 characters, 1 line".

Figure 11: AES Ciphertext

As seen in Figure 10, a key size of 128 bits was used alongside an Electronic Code Book chaining mode. AES was chosen as one encryption algorithm to test as it is widely used in corporations during encrypted communications. Furthermore, a key size of 128 bits was chosen over 192 or 256 bits as the more bits that are added, the less likely the key itself can be identified. As such, testing the 128-bit key would determine if further tests on 192- and 256-bit keys would be necessary. However, it should be noted that typically 256-bit AES keys are considered unbreakable by standard technology, and ciphertext can typically only be broken by utilising specific attacks which target vulnerabilities outside of the actual cipher [55] [56] [58] [79] [80].

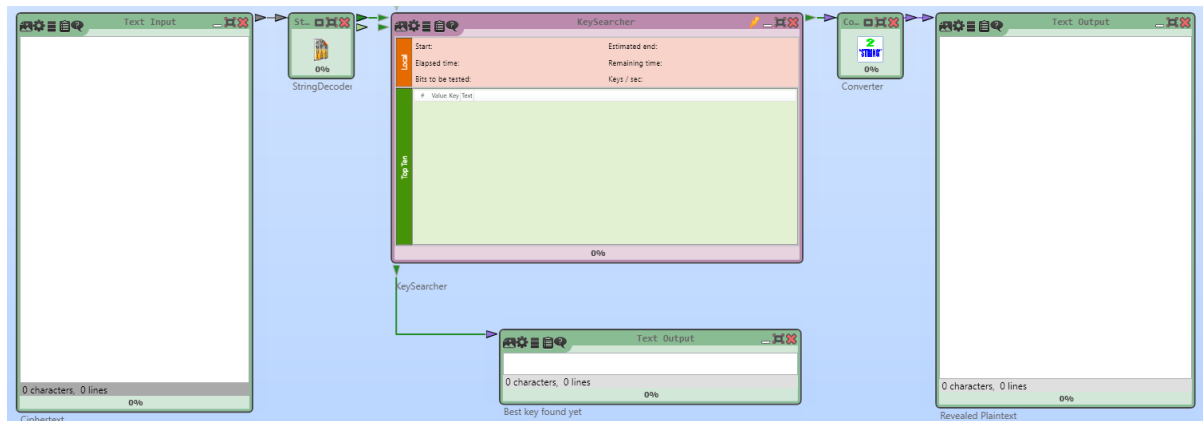


Figure 12: AES Decryption

Figure 12 shows the setup utilised in testing AES resistance to decryption. Ciphertext acquired is inserted into the left field, if processed and findings placed into the right block. A string decoder is used to ensure that only hexadecimal information is processed through the design with a converter ensuring that the results are displayed in standard ASCII text. The result demonstrates the decryption process involved.

To prove the resilience of AES at just 128-bit key length, a test was conducted where no parts of the key were known, shown in Figures 13 and 14.

Figure 13: No Known Key-bits

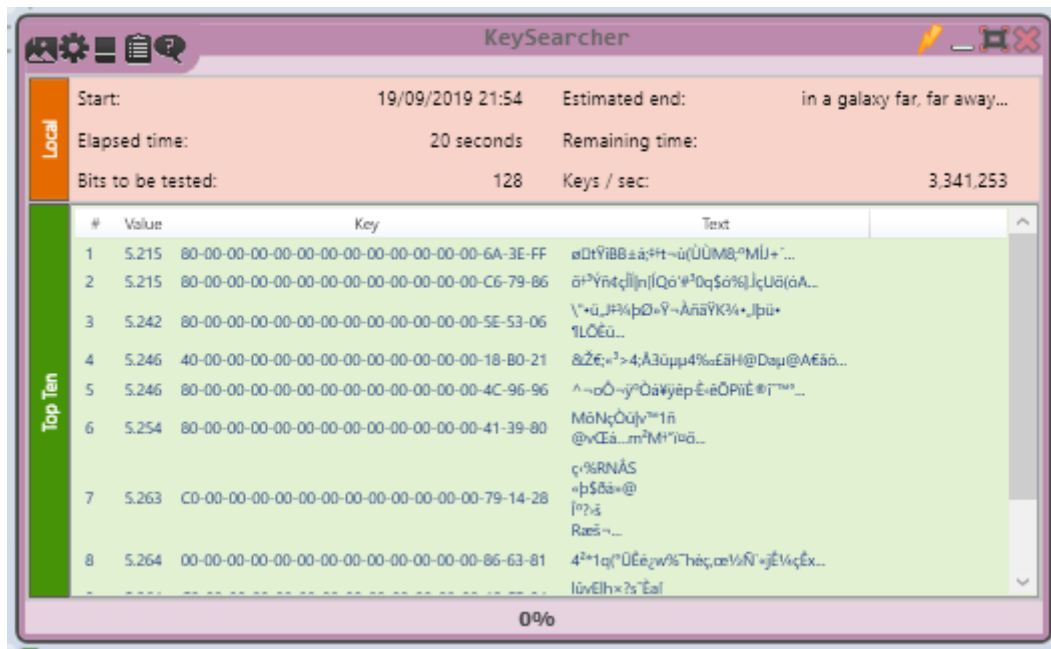


Figure 14: Test 1 Display

Figure 13 shows that no bits of the key were known, but also shows the number of cores allocated to these tests. The device performing these tests possesses an Intel Core i5 9600k processor, and all CrypTool tests will be run utilising 4 cores. As noted in Figure 14, there is no remaining time and the estimated end cannot be calculated. “In a galaxy far, far away...” is used by CrypTool to display an unknowable duration required to complete the process.

The following tests were conducted for AES:

- FD-E8-F7-A9-B8-6C-3B-FF-07-C0-D3-9D-**-**-**-**
- FD-E8-F7-A9-B8-6C-3B-FF-07-C0-D3-9**-**-**-**
- FD-E8-F7-A9-B8-6C-3B-FF-07-C0-D3-**-**-**-**
- FD-**-F7-A9-B8-**-3B-FF-07-C0-**-9D-04-60-5E-**-**
- FD-**-F7-A9-B8-**-3B-FF-**-C0-**-9D-04-60-5E-**-**

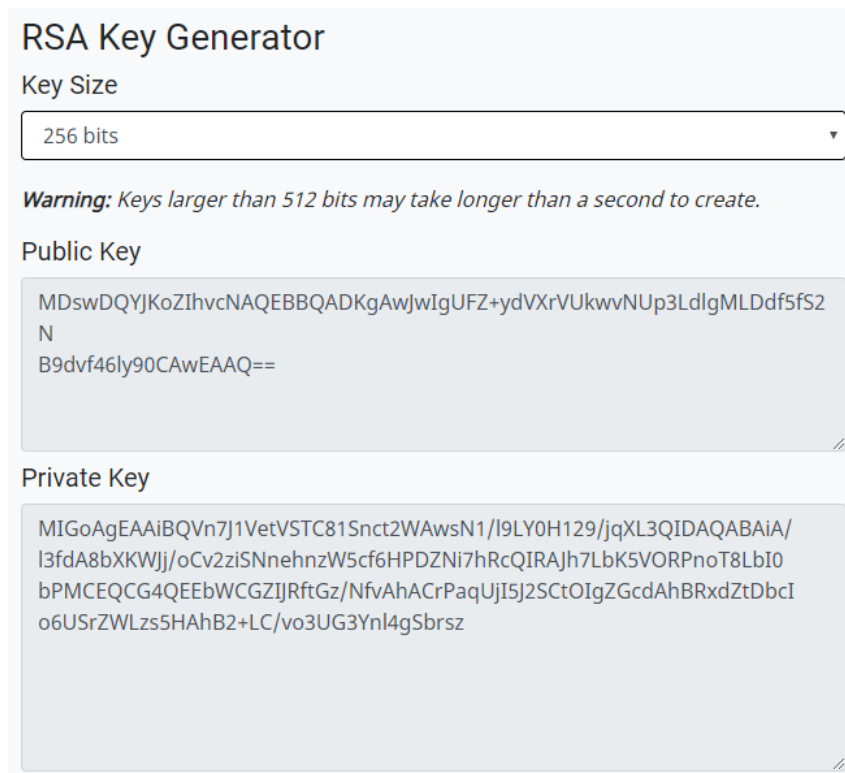
These were chosen since they can realistically be decrypted in the timespan of this project. Less and less of the actual key was utilised until it became clear that the software could not decrypt the message due to the sheer number of possibilities. The results section will cover findings and expand further upon the decision for these keys.

As the author has full knowledge of the ciphertext's original plaintext, these attacks could be considered Known Plaintext Attacks. However, the knowledge of the original plaintext is to ensure success of these attacks. In a realistic scenario, Known Plaintext attacks are almost guaranteed to fail when working with AES ciphertext. The reason for this can again be seen in Figure 14, where the time required to test every single possible key combination would be far too great. This is the primary strength of AES encryption, demonstrating its resilience to several known ciphertext attacks. Therefore, the true purpose of testing AES becomes analysing how much of the cipher key is required in order to make an attack on AES encryption possible. Acquiring portions of the cipher key can be done by utilising attacks on various vulnerabilities

outside of the AES algorithm, such as the previously mentioned Heartbleed exploit, side-channel attacks and potential BEAST and POODLE attacks.

RSA encryption algorithm is the second choice for testing purposes. This fulfils two purposes: it demonstrates testing against both symmetric and asymmetric encryption methods; and it gives an opportunity to explore vulnerabilities existing with another form of encryption. RSA is also widely utilised in different areas to AES and other symmetric encryption algorithms. AES requires less computational power and can be done on consumer-based products without sacrificing its higher security levels [55] [56] [57]. The biggest flaw is that the same key is required on both ends of communication, introducing a potential vulnerability to this encryption method [55] [56] [57] [79] [80] [101]. RSA on the hand requires more computational power and is far slower, thus it is typically used for small amounts of data [65] [66]. Its strength is that the private key to decrypt data is only known to the creator of both keys, while a public key is available to all devices participating in communication [65] [66]. RSA, therefore, relies heavily on a mathematical equation to ensure security [65] [66]. As the public key is the product of multiplying two huge prime numbers, the tests conducted will determine the maximum RSA key size that can be decrypted utilising consumer-based products.

The testing will test against 256-bit and 512-bit keys. This reason for this decision will be expanded upon shortly. Once again, the testing assumes successful acquisition of the public key. However, unlike AES, the public key is far easier to obtain than the cipher key of AES, as it is available to participants of the communication. Therefore, various attacks from Kali are capable of intercepting communication in order to obtain these public keys. As the scope of this project does not cover ethical issues with capturing data, a key generator was used to perform these tests, seen in Figure 15.



RSA Key Generator

Key Size

256 bits

Warning: Keys larger than 512 bits may take longer than a second to create.

Public Key

```
MDswDQYJKoZIhvcNAQEBBQADKgapAwJwIgUFZ+ydVXRvUkwvNUP3LdlgMLDdf5fS2
N
B9dvf46ly90CAwEAAQ==
```

Private Key

```
MIGoAgEAAiBQVn7J1VetVSTC81Snct2WAwSNI/I9LY0H129/jqXL3QIDAQABAI/
I3fdA8bXKWjj/oCv2ziSNnehnzW5cf6HPDZNI7hRcQIRAJh7LbK5VORPnoT8LbIO
bPMCEQCG4QEEbWCGZIJRftGz/NfvAhACrPaqUjI5J2SctOIgZGcdAhBRxdZtDbcI
o6USrZWLzs5HAhB2+LC/vo3UG3Ynl4gSbrsz
```

Figure 15: RSA 256-bit Key Generation

The private key aspect will be ignored for purposes of this test, as obtaining this is extremely unlikely. As seen in Figure 15, the public key would be shown in a form of code as a method of security. Within this code exists the value known as “N”, which is the multiplication total of two large prime numbers, typically labelled “p” and “q” ($N = p * q$). This code also contains a second number used for encryption and decryption purposes, labelled as “e”. In practise, e is typically set as 65,537, and is exactly this value within tests performed within this project, as will be noted later. To determine the values seen in Figure 15, a JavaScript decoder was utilised to translate the code into hex and decimal values, as seen in Figure 16.

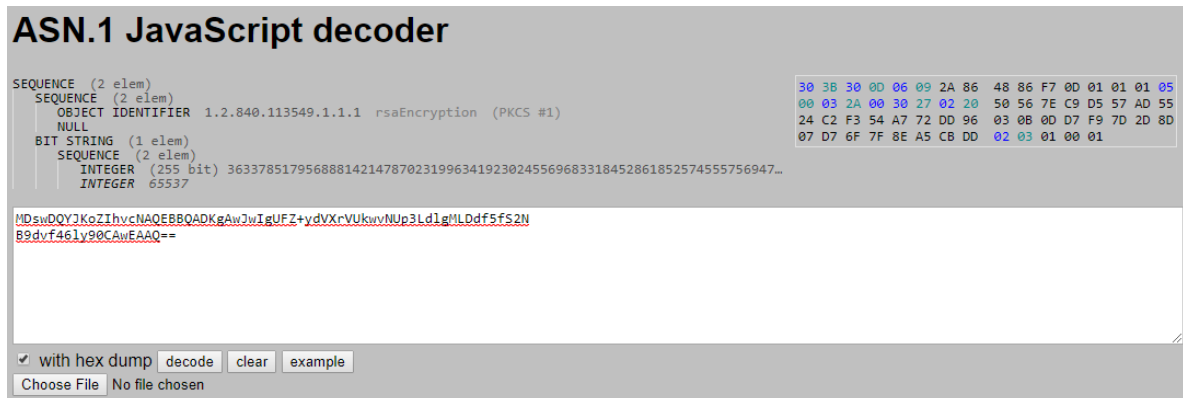


Figure 16: JavaScript decoder (256-bit key)

This gives a value to both N and e, being 36337851795688814214787023199634192302455696833184528618525745557569477659613 and 65537, respectively. These values were then used to create an encrypted message for testing purposes, seen in Figure 17.

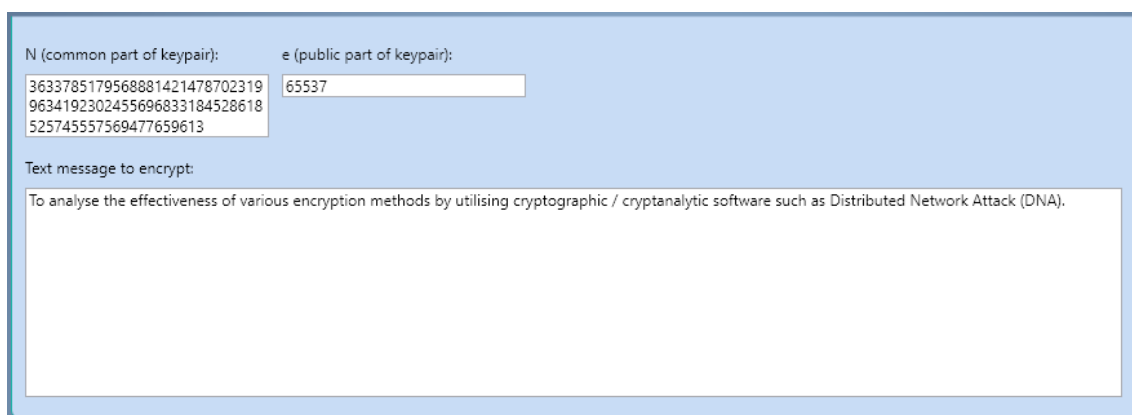


Figure 17: RSA Encryption

The same setup was utilised for an RSA 512-bit key generation, seen in Figures 18 and 19.

RSA Key Generator

Automatic
Manual

Private Key :

```

-----BEGIN RSA PRIVATE KEY-----
MIIBOQIBAAJBAISgtxVjJMI1+KvS1AzNCbKrzwrdfy16KxMQZvi946jTbwW1L/O
at81D+nWE/fh/56syRpt1A/0pup1sWYJ4oMCAwEAAQAUaeVMNC6h+usrzTOWHTA
NkW7Eeiahc0FdFb6gdwiKcmUrUuQGn1/WAJo3YU6RQ0VV12VcWg9kGJrCYav5RFG
UQIhAMtASMTN6vvIaKbCiLk/TAVGmm8rbkiGB9nMt9jBWLRLA1EApwvHSoiQQ1h7
EW009EdWhc0GaoLgPfrCwKsYe7CEd6kCIE/bWv2n/fwg2GU0p2LJHkK+LC0jfggP
YmdDVLRLX+tcTAiAjXaEzKLWTz5G8zr52LyduUmIs6IJMuhW1CK31ShGeQIgL0xt
-----END RSA PRIVATE KEY-----

```

Public Key :

```

-----BEGIN PUBLIC KEY-----
MFwwDQYJKoZIhvcNAQEBBQADSwAwSAJBASgtxVjJMI1+KvS1AzNCbKrzwrdfy1
6KxMQZvi946jTbwW1L/Oat81D+nWE/fh/56syRpt1A/0pup1sWYJ4oMCAwEAAQ==
-----END PUBLIC KEY-----

```

Generate
Key Size : 512 bit
Clear

Figure 18: RSA 512-bit Key Generation

ASN.1 JavaScript decoder

```

SEQUENCE (2 elem)
  SEQUENCE (2 elem)
    OBJECT IDENTIFIER 1.2.840.113549.1.1.1 rsaEncryption (PKCS #1)
    NULL
  BIT STRING (1 elem)
    SEQUENCE (2 elem)
      INTEGER (512 bit) 6946281184397407470242155955398921915400162544707565683216607523697994...
      INTEGER 65537

```

```

MFwwDQYJKoZIhvcNAQEBBQADSwAwSAJBASgtxVjJMI1+KvS1AzNCbKrzwrdfy1
6KxMQZvi946jTbwW1L/Oat81D+nWE/fh/56syRpt1A/0pup1sWYJ4oMCAwEAAQ==

```

☒ with hex dump
decode
clear
example

Choose File
No file chosen

Figure 19: JavaScript Decoder (512-bit key)

Once again, the e value was generated as 65537, due to this being a common number for RSA encryption, while N is set as 6946281184397407470242155955398921915400162544707565683216607523697994275703240998245479429555258560544077707650916679431757713636463020233806574762189443. Typically, the e value is set as such due to it being publicly known. Therefore, the value is inconsequential in comparison to other values, such as q , p and d . The d value is the private equivalent to the e value required for decryption. While N is utilised in an encryption process, more is required for decryption. The original q and p values must be obtained through a process called Factoring. Figure 20 shows a CrypTool design for factoring numbers, with larger numbers requiring exponentially more time.

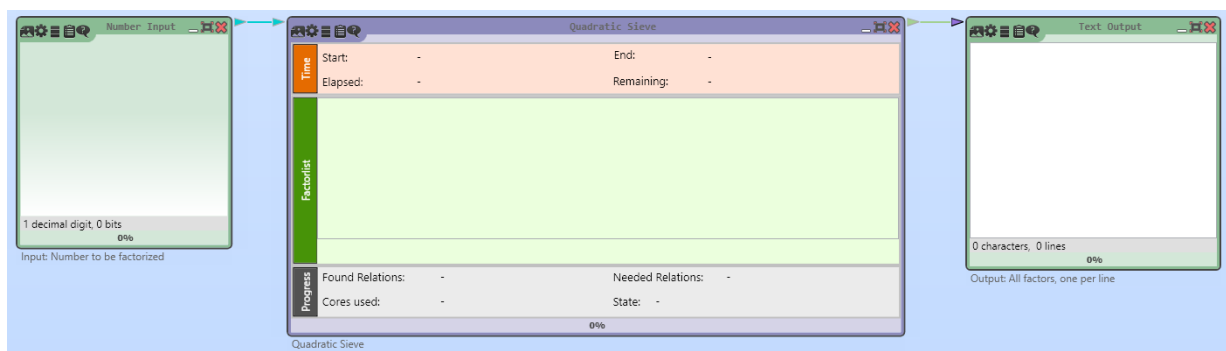


Figure 20: Factoring Design

The factoring method used is known as a Quadratic Sieve, an advantaged form of factoring for larger numbers. However, it was noted that after a certain size, numbers become too large to factor using publicly available resources. This is the reason why RSA 1024-, 2048- and 4096-bit are not tested, as it is impossible to break these with equipment available to the public. The largest known factored RSA number stands at 232 decimal digits, or 768-bit size [103].

4.2.3 Distributed Network Attack (DNA) Implementation

The last software utilised within this project was DNA. DNA is a software created by AccessData for decryption purposes. As mentioned, DNA functions by sharing cryptanalytic tasks across multiple devices [92] [93] [94]. The concept of this software is to improve time to decrypt and efficiency of decryption by utilising numerous devices in tandem. These devices would utilise their unused processing power towards the task given. As DNA is set up utilising DNA servers, clients and managers, a small group of devices would be utilised for this project to fulfil each role. Unfortunately, there were complications when attempting to utilise this software. The author was unable to obtain licensing required for utilising this software to its full potential. As this software is typically utilised for larger corporations, and as a key was not readily available for use within this project, utilising this software became impossible.

Attempts were done in order to obtain a license, including research into the software, searching online for potential answers and cost associations. Despite this, no useful information was obtained. The only possibility would be to contact AccessData directly for answers, but time restraints within the project prevented this. There is an option to utilise DNA in a demo mode. However, this too proved ineffective as the software could not identify files for decryption. Despite this, from research conducted as well as other tests run within this project, it is possible to gain an understanding of DNA's function, and possibly estimate its efficiency in comparison to other software. As DNA utilises multiple devices, it can be reasoned that it could perform these attacks in a shorter time span.

Chapter 5

5.1 Results

The implementation section discussed the setup for the testing of both AES and RSA encryption methodologies. It was identified that a pure brute-force attack is almost entirely impossible to perform as both AES and RSA keys are far too large and therefore, possess far too many possible combinations. As such, these tests are run under the assumption that previously mentioned Kali scans and attacks, as well as other potential attacks, were successful in obtaining part of an encryption key. Testing would determine the limit of breaking encryption within a realistic period. It should be noted that this project is time gated, and therefore, certain tests cannot be completed as they may require months, or even years, of run time in order to complete. These tests will still be discussed as they are still a possibility but results simply could not be recorded as the test itself would require more time than allocated to this project.

Firstly, looking at AES encryption, tests were run to determine how much of the cipher key would be required in order to decrypt a message. The first test would utilise a key value of FD-E8-F7-A9-B8-6C-3B-FF-07-C0-D3-9D-**-**-**-**. This is a relatively simple test as most of the key is known, and as such, should not require an extraordinary amount of time.

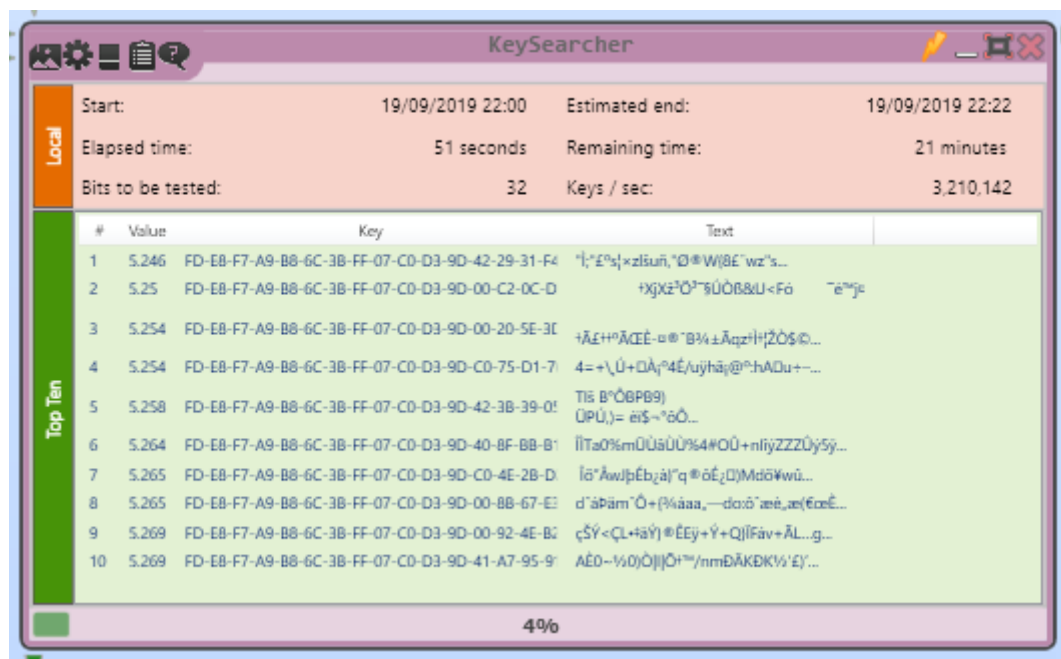


Figure 21: First AES Test Run Timer

As seen in Figure 21, the software would test all possible combinations in order to obtain the remainder of the cipher key. As most of the key was known, an estimated time of 21 minutes was given.

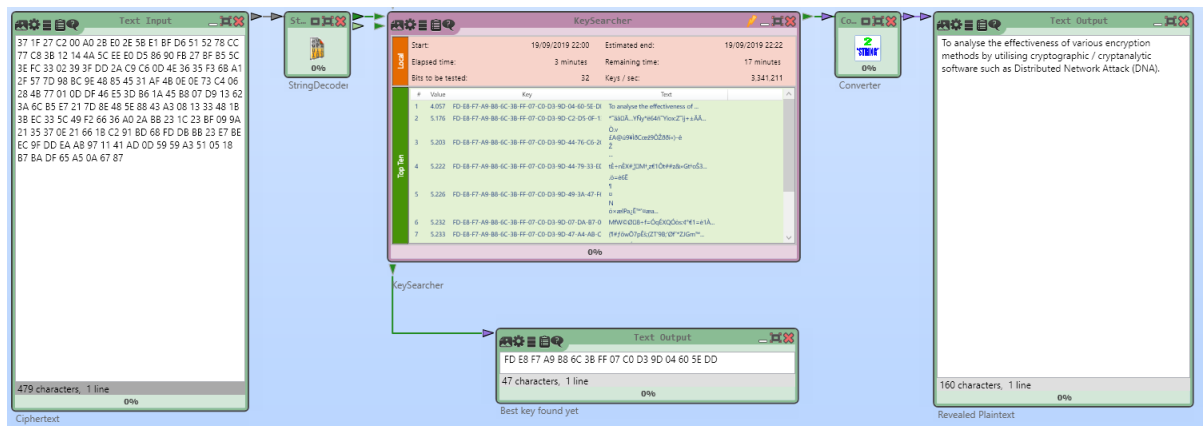


Figure 22: First AES Test Completion

Figure 22 shows the completion of the first test run. Despite Figure 21 showing a possible run time of 21 minutes, the test itself successfully retrieved the key within 3 minutes. This is due to the 21 minutes signifying the total time required to test all possible combinations. As the author knew the contents of the ciphertext, it was possible to mark the actual completion timer. In a real-world scenario, the ciphertext may be unknown to the attacker. Therefore, a real scenario test would be run for the entire duration.

Follow up tests were conducted with less and less of the cipher key. These tests were run with FD-E8-F7-A9-B8-6C-3B-FF-07-C0-D3-9*-*-*-*-* and FD-E8-F7-A9-B8-6C-3B-FF-07-C0-D3-*-*-*-*-*.*. As seen, less bits were known within these tests and the results can be seen below.

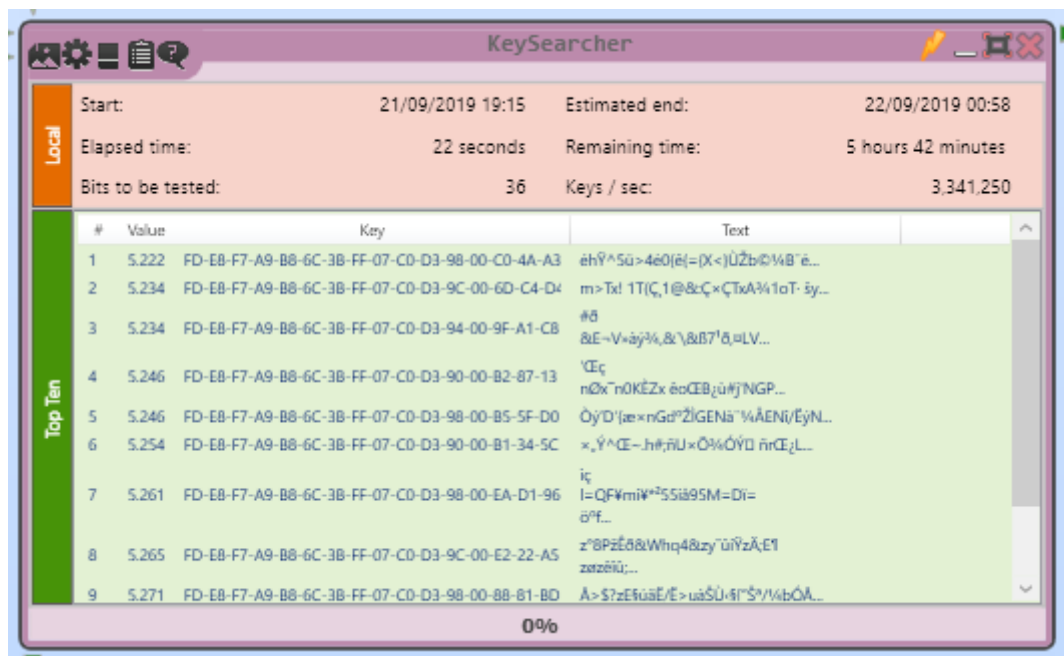


Figure 23: Second AES Test Run Timer

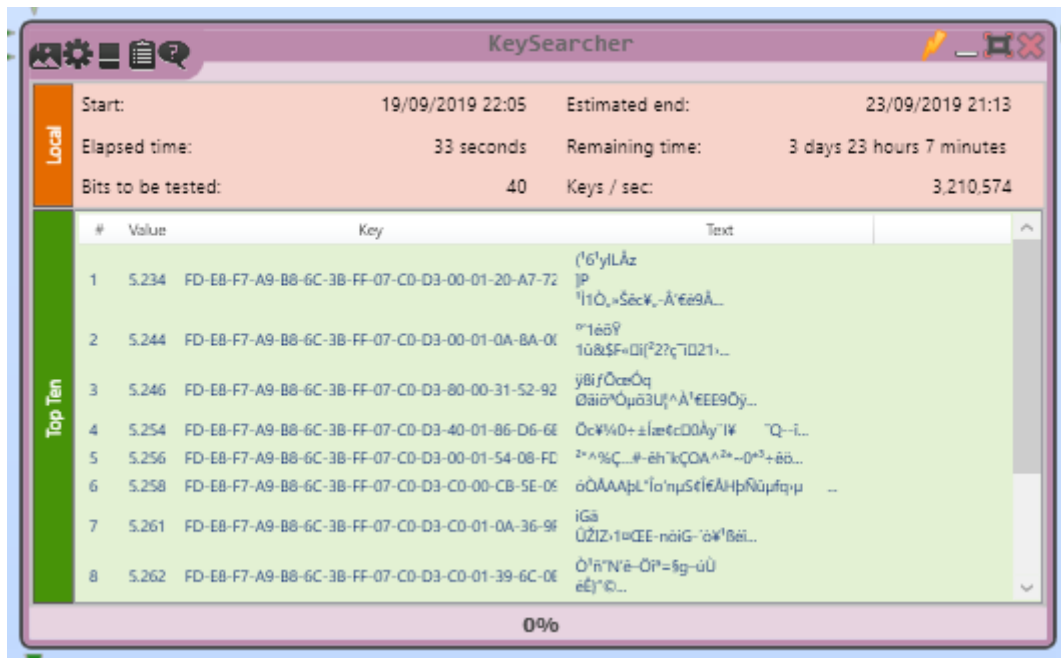


Figure 24: Third AES Test Run Timer

Figures 23 and 24 show the second and third test runs times respectively. The first thing worth noting is the increased time required for test completion, in comparison to the first test. A time of 5 hours and 42 minutes is required for the second test to complete. The key utilised in the second test possessed 4 less bits than in the first. This led to a drastic increase in time required to test all possible combinations. During the third test, shown in Figure 24, a key with 4 less bits once more was used. Again, the results show a drastic increase in time for completion. Figure 24 shows a duration of just under 4 days to complete. One keynote, however, is that this is simple to attempt every possible combination within the key. Figures 25 and 26 show a completed run for the second test.

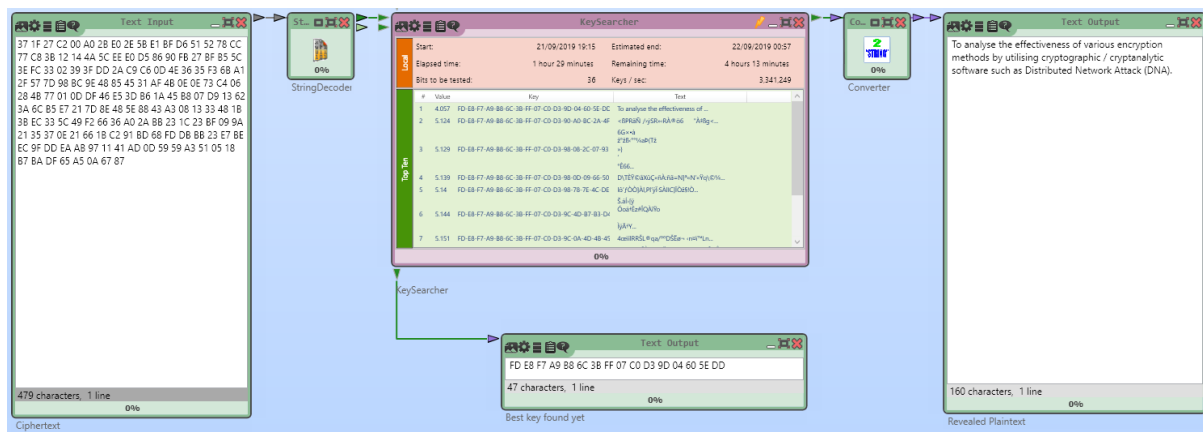


Figure 25: AES Second Test Completion

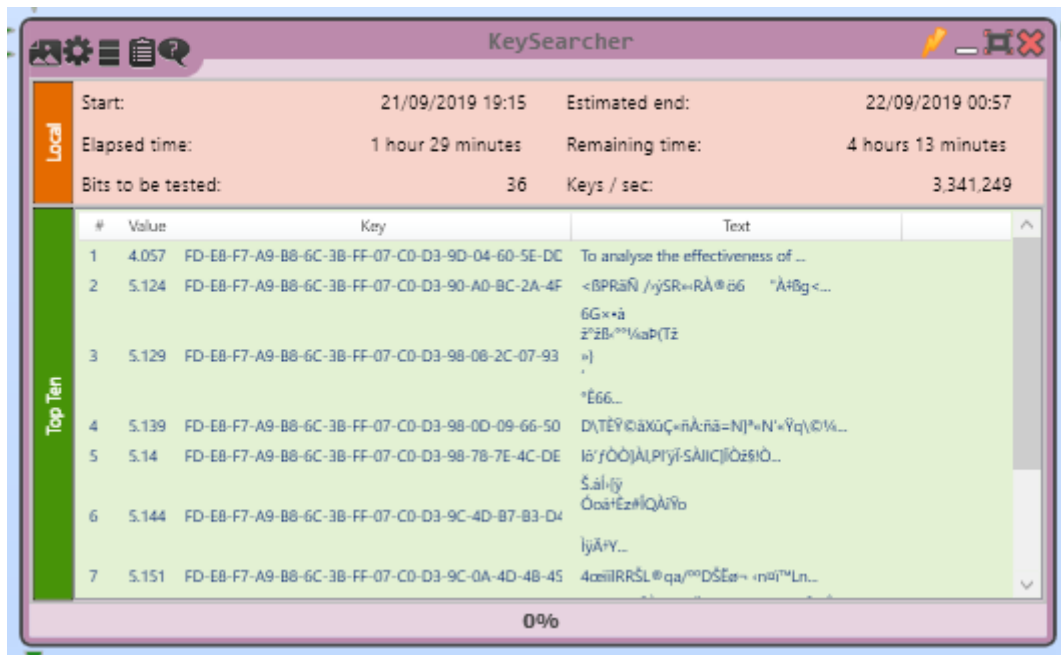


Figure 26: AES Second Test Completion Time

Originally it was stated that the test would require a total of 5 hours and 42 minutes to complete. Yet, Figure 26 shows the key was isolated within 1 hour and 29 minutes. This is once again because the completion time is simply to attempt every possible combination, not to obtain the correct key. In a real-world scenario, it can be difficult to determine if the key is obtained early, and it is due to the authors knowledge of the plaintext content that aids in determining the true time taken for each test. As such, these tests would typically be run for the entire duration.

When looking back at an original attempt where none of the key was known, and the follow up tests where much of the key was known, it becomes clear that cracking AES is heavily dependent on a previous step in implementation. That is the scanning and vulnerability attack step, demonstrated in this project utilising Kali. This coincides with research into AES [55] [56] [57] and into other cryptographic vulnerabilities [79] [80] [101]. These results also reinforce previous research regarding AES capabilities of resisting numerous cryptanalysis attacks.

Further tests involved removing more and more of the known key bits to examine the time requirement. However, as expected, the time required in order to isolate the key dramatically increased to a point that exceeded this projects time span. Despite this, it is important to understand the impact each bit of an AES key has upon the algorithm's security. This, in combination with its function of performing encryption in rounds, allow AES encrypted information to remain confidential, with only outside vulnerabilities being this algorithm's true weakness. Additional tests were run utilising the following keys:

- FD-**-F7-A9-B8-**-3B-FF-07-C0-**-9D-04-60-5E-**-
- FD-**-F7-A9-B8-**-3B-FF-**-C0-**-9D-04-60-5E-**-

These tests are shown in Figures 27 and 28.

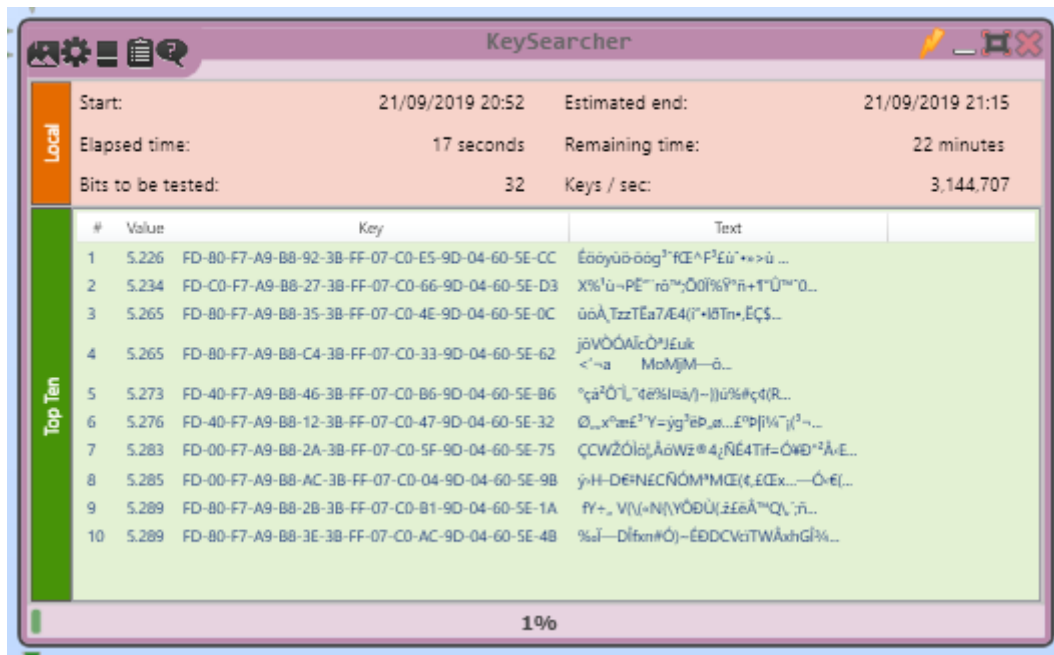


Figure 27: AES Fourth Test

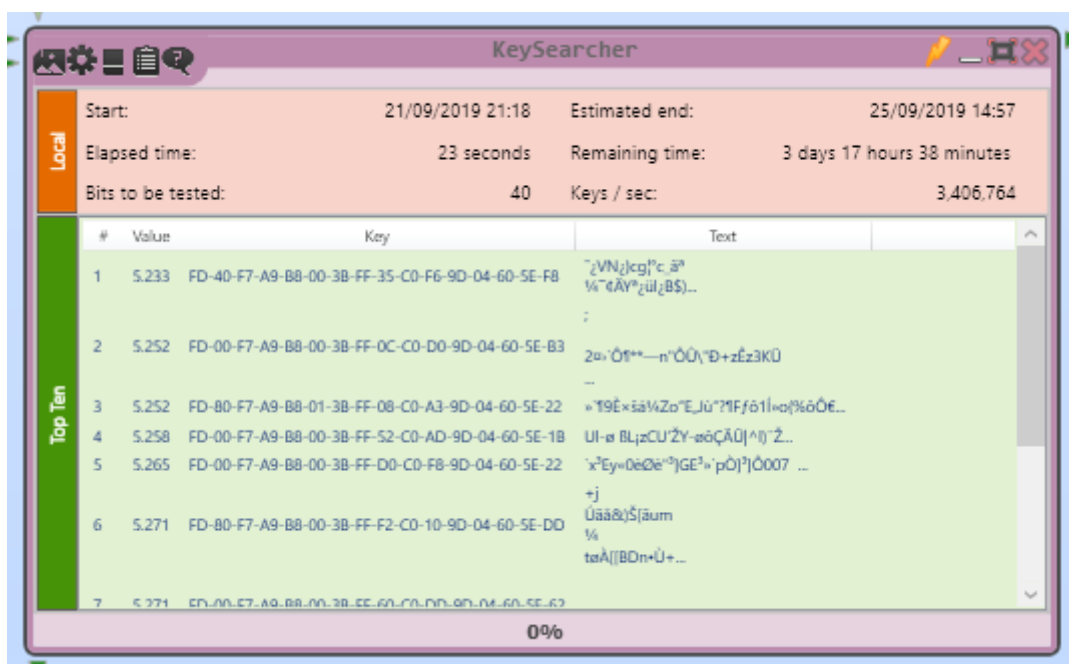


Figure 28: AES Fifth Test

These tests shown in Figures 27 and 28 were mainly to demonstrate that the location of each unknown key bit did not affect the test duration. The test shown in Figure 27 is like that shown in Figure 21, while the test shown in Figure 28 is like the one shown in Figure 26. As noted, the time required to complete each test remained the same, despite the shuffling of unknown bits.

Despite the inability to utilise DNA to perform similar tests utilising the same keys, by examining what information is available regarding this software, as well as understanding the computational power required to perform cryptanalysis attacks, it can be theorised that DNA

would be capable of performing these tests more efficiently. This is due to DNA allocating bits of a key to various machines, enabling multiple devices to cooperate on a single task of decryption. Therefore, despite not knowing precisely how much time could be saved utilising DNA, and analysing the time required for a single machine to perform similar attacks, it can be estimated that DNA enabled systems could acquire AES keys faster, assuming each device is running equivalent hardware. This also lends credit to DNA systems being capable of decrypting AES ciphertext when even less of the key is known to the attacker. However, regarding research into AES, this hypothesis may only go so far as AES has been known to be extremely challenging to combat within the cryptanalysis communities.

Following tests utilising AES encryption, several tests using RSA were conducted. As mentioned, a formula exists in the creation of private and public keys used for decryption and encryption processes. The values for both N and e are available within the public key and can be seen in Figures 16 and 17. As mentioned, for decryption to be possible, the values for q and p must be retrieved by factoring the value of N. Figure 18 showed the design for this while Figure 29 shows the results of factoring the N value for an RSA 256-bit key.

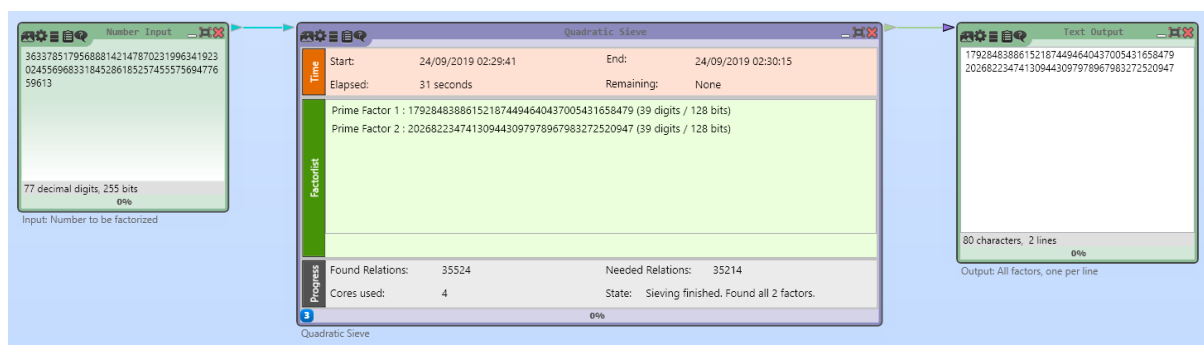


Figure 29: Factoring N Value of RSA 256-bit Key

As seen in Figure 29, 2 numbers were identified by factoring the N value: 36337851795688814214787023199634192302455696833184528618525745557569477659613. Factoring refers to a mathematical process where a number is broken into smaller numbers, which, if are multiplied together, create the original number. An example of this would be breaking 6 into 3 and 2, as these two new numbers can be multiplied to retrieve the original. The two new values obtained through factoring the value of N are known as p and q, which are 179284838861521874494640437005431658479 and 202682234741309443097978967983272520947, respectively. These values are seen above in Figure 29, and below in Figure 30.



Figure 30: RSA 256-bit Key p and q Values

The reason that only these numbers are shown, despite there being numerous numbers that can be multiplied to retrieve the original, is because these are known as Prime Factors. Prime Factors are essentially Prime numbers that happen to be factors of a greater number. Prime numbers are numbers greater than 1 that cannot be made by multiplying smaller whole numbers. The values given to p and q cannot be formed by multiplying smaller numbers, and it is these values that are used to contribute to RSA's private key generation.

The remainder of the formula involved in creating a private key becomes increasingly complicated. This is further reinforced when dealing with numbers as large as ones utilised in RSA encryption. The formula [102] used in the creation of a private key is:

- $d = 1/e \bmod \lambda(n)$

where d is the secret value known only to the creator of the RSA key. This value is critical in order to decrypt RSA ciphertexts. The above formula, in conjunction with another formula [102]:

- $m = c^d \bmod n$

are the functions that operate within the background of the CrypTool operation used for this project. As RSA numbers are typically extremely large, it becomes impossible to perform these operations without the assistance of various software. However, larger RSA keys are impossible to break in this method utilising current available technology. The largest known key to be deciphered was an RSA 768-bit key [103].

Returning to the test results of this project, now that values of N, e, p and q are discovered, it is possible to decrypt the initial message, seen back in Figure 17.

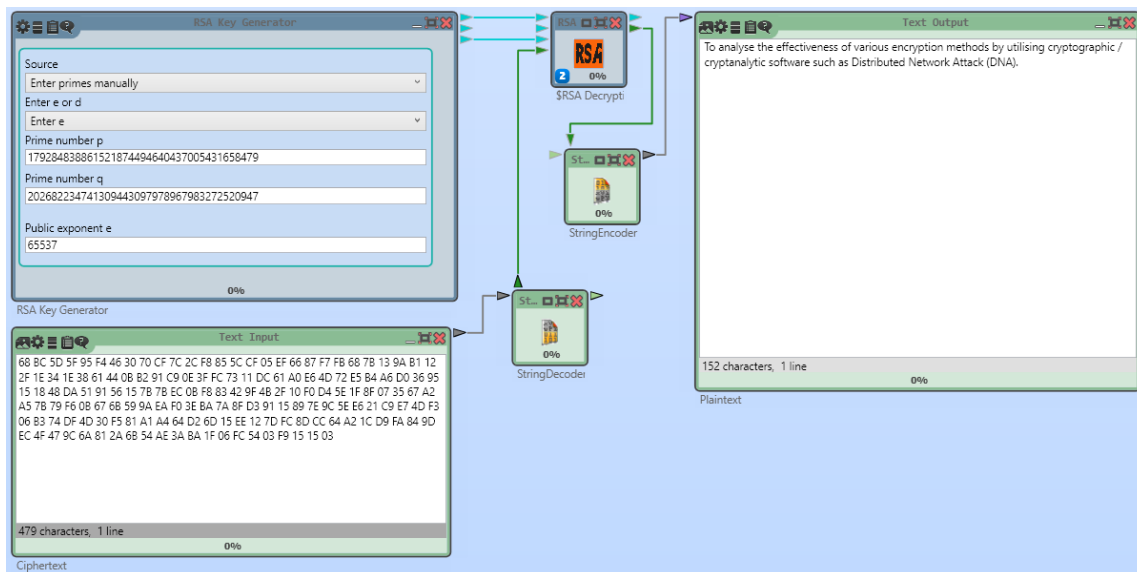


Figure 31: RSA 256-bit Decryption

Figure 31 shows the setup used to decrypt RSA encryption. The two previously mentioned formulas are utilised by this software to recreate the private key in order to decrypt the message, seen in Figure 31. Once the value of N has been factored, RSA private keys can be identified. The problem, however, exists in factoring the N value to begin with. A test was run utilising an RSA 512-bit key. The value given to N in this scenario was far larger than that of a 256-bit key, and as such required far more time to factor. Factoring a 512-bit key is possible and was first done back in 1999 [104]. However, factoring a number this large utilising software and equipment available to the author required longer than the time allocated for this project. The process and theory remain precisely the same as the previous test but would require far more processing power and time. As a key size of 768-bit is the largest known RSA key to be broken [103], it can safely be assumed that higher keys are far beyond the scope of this project.

Once again, as DNA was not available for testing RSA decryption it is unknown how the software would fare. As RSA cryptanalysis attacks are vastly different to AES, as this algorithm is an asymmetric design, it is unknown if DNA would even be capable of deciphering this method of encryption. The main unknown is if DNA can perform factoring procedures, as this is the main obstacle to overcome regarding RSA cryptanalysis.

To conclude the results of this project, both AES and RSA have shown to be very secure against many cryptanalysis attacks. AES vulnerabilities exist more within its implementation into a system, rather than as an actual flaw within the algorithm itself. As such, attacks that exploit side-channel vulnerabilities, or attacks such as Heartbleed, POODLE and BEAST prove to be the most effective manner of circumventing AES encryption. On the other hand, RSA vulnerabilities also exist in side-channel exploits, but also in man-in-the-middle (MitM) attacks to intercept traffic during the formation of public and private keys. As the latter is an unreliable expectation, the main obstacle in circumventing RSA security returns to the need for greater software and technology in order to perform factoring upon larger numbers. Due to this reason, it is believed that RSA methods of encryption are more secure in comparison, but as it requires more computation power than AES, and AES itself is also highly secure, a case can be made in favour of AES encryption.

Chapter 6

6.1 Conclusion

The purpose of this project was to analyse various cryptographic implementations in order to identify their strengths and weaknesses. Following this, various cryptanalysis methods were utilised to break these ciphertexts in order to obtain the original plaintext. Firstly, research was done in order to gain a greater understanding of cryptography, cryptanalysis and various security functions related to this topic. Security is a vast topic which covers numerous areas within networking. Furthermore, the field itself is constantly changing as new iterations of technology are developed and made available for public use. Despite this, few concepts remain unchanged. An example of these are the pillars of internet security: confidentiality, integrity and availability [7] [8] [9] [11]. These are the pillars upon which cryptology, and its subfields cryptography and cryptanalysis, are built upon. Not only this field, but security in general is heavily defined by these requirements.

Confidentiality refers to protection of data, specifically, ensuring that data is not exposed to any unauthorised individuals or groups [7] [8]. Integrity ensures that data is not tampered with during its lifespan [8] [9]. This means that data must not be changed, instead, it must always remain a constant. Finally, availability refers to ensuring that data may always be accessed by authorised individuals or groups [11]. This ensures that network resources, personal information and any other form of data is always accessible. These three are notable for security in general, but each play a part within cryptography. As information goes through a process of alteration into a code, or ciphertext, unauthorised groups are unable to access the original data, thus ensuring confidentiality. Despite alterations into code, the content of the data does not get altered or changed. Rather, the data is simply displayed in an alternate form which can only be understood by those possessing a special key used to obtain the data's original form. While availability is the least utilised pillar regarding cryptography, it is still crucial as access to data at any point will constantly remain a priority for organisations and enterprises.

Through the literature review, two main forms of cryptography were identified: symmetric and asymmetric encryptions [13] [52] [53]. Special keys are utilised in order to encrypt and decrypt information. However, the function of these keys differs between symmetric and asymmetric designs. With symmetric encryption, the same key is utilised for both encryption and decryption purposes [13] [52] [53]. However, asymmetric designs use a public key to encrypt messages and a private key to decrypt [13] [52] [53]. There are many ciphers available utilising these designs. Through extensive research, the main methods identified are AES [56] [57] [58], DES [58] [60] [61] and RSA [65] [66]. The first two are categorised as symmetric ciphers, while RSA is classed as an asymmetric cipher. These are the main ciphers focused on during the project as it gives an example of both main categories, and due to their wide implementation, an understanding of their strengths and weaknesses would be beneficial.

However, simply identifying these methods of cryptography were not enough for this project needs. Further research into cryptanalysis was conducted in order to identify potential weaknesses involves with these cryptosystems. Certain attacks against ciphers were identified, such as: chosen plaintext, known plaintext, ciphertext-only attacks and, in for RSA specifically, factorisation [81] [82] [83]. Other attacks were identified against aspects of the cipher's implementation. These vulnerabilities included differential analysis, side-channel attacks and Heartbleed vulnerabilities [79] [80] [85] [101]. A number of these attacks were decided upon

due to the variance in vulnerabilities, i.e. some targeting the cipher itself while others target the implementation. This would give a good foundation for determining each ciphers security level.

The software chosen for this project were: Kali Linux [100], which was run using the virtual machine software Oracle VM; CrypTool 2 [99], a cryptography and cryptanalysis tool possessing numerous cryptanalysis functions and capabilities; and Distributed Network Attack (DNA) [91] [92] [93], a software developed by AccessData [95] for the purposes of advanced decryption. DNA functions by distributed workload across multiple machines. This effectively enables numerous devices to work on a single task of decryption, improving efficiency and time requirement.

Scans were run utilising Kali, including those needed for Heartbleed exploits, BEAST and POODLE attacks, as well as other side-channel and differential attacks. Further tests were run using CrypTool to determine each ciphers resistance to various cryptanalysis attacks. Unfortunately, DNA was unable to be utilised due to unforeseen technical issues. Despite many workarounds being deployed, the software could not be setup for use within this project. This will be discussed further within the Future Works section following this conclusion. However, DNAs effectiveness can be estimated based on test results run within this project and information obtained from research.

While testing AES, it became clear that this cipher is very resilient to standard cryptanalysis attacks. This is attributed to its key size and to the function in which it encrypts data. AES performs rounds of encryption, with additional rounds for larger key sizes. This affords AES with high levels of security. Tests were run where various parts of an AES were unknown. Results of this show that an entirely unknown key cannot be deciphered with the devices used within this project. Testing with various levels of the key known to the attacker showed dramatic changes in time requirement. With simply 24-bits unknown, these attacks required roughly 21 minutes to execute. However, removing a further 4-bits extended this time requirement to roughly 6 hours. A further 4-bits were subsequently removed, bringing the total unknown to 32-bits, which increased attack time required to 4 days. This level of time increase continued at this pace until it was no longer possible to identify a time frame for these attacks.

Cross-referencing this with related works indicate AES may not be possible to break using standard cryptanalysis attacks. As such, other vulnerabilities were tested. Chief among these was observing potential Heartbleed vulnerabilities and Side-channel vulnerabilities. Due to the limitations of this project, major attacks were not launched. Instead scans were conducted to signify if a device was vulnerable, while using related works to discuss further steps of these attacks.

Moving on to RSA testing, different procedures were required as this is an entirely different method of cryptographic design. RSA is an asymmetric cipher which uses complicated mathematical equations in conjunction with large numbers to ensure data security. As such, the main method of attack upon RSA was the factoring attack, using msieve factorisation. This is a method of factoring larger than standard numbers. Factoring refers to separating a single number into two smaller numbers. These numbers must be capable of recreating the original by means of multiplication. It was identified that RSA keys, both private and public, utilise prime factors within their equations. Therefore, factor attacks against RSA keys look for specific numbers in order to recreate the private key, and therefore, decrypt ciphertext. Tests

were run against 256- and 512-bit keys. Higher keys were not done as the largest known RSA key broken was 768-bit [103]. This would require more time than is afforded to this project. However, simply observing the results of 256- and 512-bit keys shows a large increase in time requirement. 256-bit RSA was successfully broken and utilising the same concept against 512-bit required far more time. Research indicates that utilising current publicly available technology is not enough to factor numbers associated with 1024-, 2048- and 4096-bit RSA keys. Hence these keys were not utilised for testing purposes within this project.

Based on results of both AES and RSA testing, it was identified that RSA possesses higher security levels in comparison to AES. However, it should be noted that AES security levels are exceedingly high, and this method requires far less computational power than RSA, making it an ideal choice for many groups. The observation of RSA being greater than AES does not come from time requirement to decrypt or factor cipher keys, but since AES possesses more vulnerabilities in its setup, which allows for more possible vectors of attack in comparison to RSA.

6.2 Future Work

Should a scenario arise where this project is repeated, a few changes would be done in order to improve upon the results. Firstly, while it is possible to theorise DNA's effectiveness, setting the software up and having it run without any issues would solidify any hypothesis regarding its efficiency. This change would reinforce any findings regarding each cipher's security, as well as gaining a greater understanding of DNAs capabilities.

Secondly, additional ciphers would be used throughout the project. AES and RSA were chosen due to their popularity in modern systems participating in encrypted communication. However, these ciphers are also the most resistant to certain cryptanalysis attacks. This is both a positive and negative feature in testing. A positive note would present itself should an unknown security flaw be presented during testing, furthering research related to said cipher. The negative, however, involves having limited research to draw upon regarding testing various attacks. As these ciphers are very resilient to modern cryptanalysis attacks, with their main vulnerability being in the systems they are implemented into, this leaves little room for attack variety. Most tests run within this project assumed a successful Kali attack, but this scenario cannot be guaranteed in a real-world scenario.

Third, a case could be made in identifying a target which utilises the ciphers tested within this project. If it were possible to identify such a system and obtain permission to pen test this system in an attempt to bypass cryptographic security measures and obtain data stored within a network or system, then results would become more realistic in a sense that they would reflect a real-world scenario more accurately. This was not done for this project as this weakness did not become as apparent until late into the project's life cycle. As such, this is one area that the author would like to improve upon to solidify the results found.

Further avenues exist for future work regarding this topic. As mentioned during the introduction of this project, an original goal existed to test if it is possible to mine cryptocurrency utilising available computing capacity within the deep/dark web using Distributed Network Attack (DNA). This project presented a start which could be expanded in order to test this previous hypothesis. The area of cryptography and cryptanalysis has now been

covered, with additional research to get DNA operational, followed by research into the deep/dark web and cryptocurrency, could lead to a decisive answer for this question.

Utilising research found within the literature review of this project, followed by tests run, lead the author to believing that it is not possible to achieve this goal in this manner. This is due to the nature of DNA and that of the deep/dark webs. DNA requires a DNA server, which is installed on a device located in a central accessible area of the network, client devices in which tasks can be assigned and a DNA manager which would be used to coordinate these attacks [92] [93] [94]. Hypothetically, if one were able to set up these systems across devices connected to the deep and / or dark web, they would still only utilise their own resources to perform these attacks. This is because DNA clients are the devices to orchestrate a DNA attack, with coordination from a DNA manager device. As such, deep/dark web computing capacity would not be utilised, rather, it would simply act as a form of communication for devices participating in this DNA attack. Due to the anonymity granted utilising the dark web, this could be advantageous in certain scenarios. However, it is unlikely that deep/dark web resources could be used to mine cryptocurrency using just DNA.

This is simply the authors interpretation of how DNA functions. To prove this hypothesis, future work would need to be conducted in the fields of cryptocurrency, the deep web and the dark web. Perhaps supplementary software could be identified which could work in conjunction with DNA to mine cryptocurrency. Or perhaps DNA has additional functions not yet identified by the author which could allow it to be used in this capacity. In a scenario where this topic is expanded upon, the author would conduct research into these previously mentioned fields, as well as further research into DNA itself, in order to identify tests which would lead to proving or disproving the hypothesis that the original goal cannot be achieved in the manner presented. It should be noted that this thought process is entirely theoretical, based upon research done on DNA alone. The goal itself may be very possible but would require additional time, resources and research into fields to determine possible outcomes.

References

- [1] H. Ge, L. Gu, Y. Yang and K. Liu, "An attack graph based network security evaluation model for hierarchical network," *2010 IEEE International Conference on Information Theory and Information Security*, Beijing, 2010, pp. 208-211.
- [2] X. Li, L. Yu and L. Wei, "The application of hybrid encryption algorithm in software security," *2013 3rd International Conference on Consumer Electronics, Communications and Networks*, Xianning, 2013, pp. 669-672.
- [3] W. Kehe, Z. Tong, L. Wei and M. Gang, "Security Model Based on Network Business Security," *2009 International Conference on Computer Technology and Development*, Kota Kinabalu, 2009, pp. 577-580.
- [4] F. Lan, W. Chunlei, M. Qing and L. Li, "Dynamically validate network security based on adaptive control theory," *2013 International Conference on Information and Network Security (ICINS 2013)*, Beijing, 2013, pp. 1-6.
- [5] S. V. Thakare and D. V. Gore, "Comparative Study of CIA and Revised-CIA Algorithm," *2014 Fourth International Conference on Communication Systems and Network Technologies*, Bhopal, 2014, pp. 713-718.
- [6] P. Papadimitriou and H. Garcia-Molina, "A Model for Data Leakage Detection," *2009 IEEE 25th International Conference on Data Engineering*, Shanghai, 2009, pp. 1307-1310.
- [7] K. Keerthi and B. Surendiran, "Elliptic curve cryptography for secured text encryption," *2017 International Conference on Circuit ,Power and Computing Technologies (ICCPCT)*, Kollam, 2017, pp. 1-5.
- [8] J. Yin and D. Zhao, "Data confidentiality challenges in big data applications," *2015 IEEE International Conference on Big Data (Big Data)*, Santa Clara, CA, 2015, pp. 2886-2888.
- [9] A. N. Rukavitsyn, K. A. Borisenko, I. I. Holod and A. V. Shorov, "The method of ensuring confidentiality and integrity data in cloud computing," *2017 XX IEEE International Conference on Soft Computing and Measurements (SCM)*, St. Petersburg, 2017, pp. 272-274.
- [10] R. C. Joseph, "Data Breaches: Public Sector Perspectives," in *IT Professional*, vol. 20, no. 4, pp. 57-64, Jul./Aug. 2018.
- [11] S. Kang, B. Veeravalli, K. M. M. Aung and C. Jin, "An efficient scheme to ensure data availability for a cloud service provider," *2014 IEEE International Conference on Big Data (Big Data)*, Washington, DC, 2014, pp. 15-20.
- [12] M. Soliman and M. A. Azer, "Web Application API Blind Denial of Service Attacks," *2018 14th International Computer Engineering Conference (ICENCO)*, Cairo, Egypt, 2018, pp. 249-253.
- [13] M. Moizuddin, J. Winston and M. Qayyum, "A comprehensive survey: Quantum cryptography," *2017 2nd International Conference on Anti-Cyber Crimes (ICACC)*, Abha, 2017, pp. 98-102.

- [14] B. Akiwate and L. Parthiban, "A Dynamic DNA for Key-based Cryptography," *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*, Belgaum, India, 2018, pp. 223-227.
- [15] M. F. B. Rafiuddin, H. Minhas and P. S. Dhubb, "A dark web story in-depth research and study conducted on the dark web based on forensic computing and security in Malaysia," *2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, Chennai, 2017, pp. 3049-3055.
- [16] A. T. Zulkarnine, R. Frank, B. Monk, J. Mitchell and G. Davies, "Surfacing collaborated networks in dark web to find illicit and criminal content," *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, Tucson, AZ, 2016, pp. 109-114.
- [17] S. Ghimire and D. H. Selvaraj, "A Survey on Bitcoin Cryptocurrency and its Mining," *2018 26th International Conference on Systems Engineering (ICSEng)*, Sydney, Australia, 2018, pp. 1-6.
- [18] A. Rajendra Sai Lero, J. Buckley Lero and A. Le Gear, "Privacy and Security Analysis of Cryptocurrency Mobile Applications," *2019 Fifth Conference on Mobile and Secure Services (MobiSecServ)*, Miami Beach, FL, USA, 2019, pp. 1-6.
- [19] FangLan, Wang Chunlei, MiaoQing and LiuLi, "Dynamically validate network security based on adaptive control theory," *2013 International Conference on Information and Network Security (ICINS 2013)*, Beijing, 2013, pp. 1-6.
- [20] Y. Wang, J. Lu, Z. Wu and Y. Lu, "Component Based Security Control for Information Network," *The Proceedings of the Multiconference on "Computational Engineering in Systems Applications"*, Beijing, 2006, pp. 1357-1360.
- [21] IEEE/ISO/IEC Information technology -- Telecommunications and information exchange between systems -- Local and metropolitan area networks -- Part 1X: Port-based network access control," in *ISO/IEC/IEEE 8802-1X:2013(E)* , vol., no., pp.1-228, 6 Dec. 2013
- [22] C. Xue, X. Wu, T. Qin, H. Bi and Z. Wang, "Improvement of ACL assembly algorithm," *2017 2nd International Conference on System Reliability and Safety (ICSRS)*, Milan, 2017, pp. 380-385.
- [23] M. Naveed, S. un Nihar and M. Inayatullah Babar, "Network intrusion prevention by configuring ACLs on the routers, based on snort IDS alerts," *2010 6th International Conference on Emerging Technologies (ICET)*, Islamabad, 2010, pp. 234-239.
- [24] K. Yoshizaki and T. Yamauchi, "Malware Detection Method Focusing on Anti-debugging Functions," *2014 Second International Symposium on Computing and Networking*, Shizuoka, 2014, pp. 563-566.
- [25] S. Khummanee, A. Khumseela and S. Puangpronpitag, "Towards a new design of firewall: Anomaly elimination and fast verifying of firewall rules," *The 2013 10th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Maha Sarakham, 2013, pp. 93-98.
- [26] Chenghong Wang, Donghong Zhang, Hualin Lu, Jing Zhao, Zhenyu Zhang and Zheng Zheng, "An experimental study on firewall performance: Dive into the bottleneck for firewall

effectiveness," *2014 10th International Conference on Information Assurance and Security*, Okinawa, 2014, pp. 71-76.

[27] J. Tian and M. Gao, "Network Intrusion Detection Method Based on High Speed and Precise Genetic Algorithm Neural Network," *2009 International Conference on Networks Security, Wireless Communications and Trusted Computing*, Wuhan, Hubei, 2009, pp. 619-622.

[28] Y. Yang, P. Zeng, X. Yang and Y. Huang, "Efficient Intrusion Detection System Model in Wireless Mesh Network," *2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing*, Wuhan, Hubei, 2010, pp. 393-396.

[29] A. Kavianpour and M. C. Anderson, "An Overview of Wireless Network Security," *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*, New York, NY, 2017, pp. 306-309.

[30] J. Zhang, "Research on Key Technology of VPN Protocol Recognition," *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, Chongqing, China, 2018, pp. 161-164.

[31] C. Fei, W. Kehe, C. Wei and Z. Qianyan, "The Research and Implementation of the VPN Gateway Based on SSL," *2013 International Conference on Computational and Information Sciences*, Shiyang, 2013, pp. 1376-1379.

[32] F. Duchène, S. Rawat, J. Richier and R. Groz, "LigRE: Reverse-engineering of control and data flow models for black-box XSS detection," *2013 20th Working Conference on Reverse Engineering (WCRE)*, Koblenz, 2013, pp. 252-261.

[33] A. Tetskyi, V. Kharchenko and D. Uzun, "Neural networks based choice of tools for penetration testing of web applications," *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kiev, 2018, pp. 402-405.

[34] K. C. Purohit, S. Bisht, A. Joshi and J. Bhatt, "Hybrid approach for securing IoT communication using authentication and data confidentiality," *2017 3rd International Conference on Advances in Computing, Communication & Automation (ICACCA) (Fall)*, Dehradun, 2017, pp. 1-6.

[35] Tomáš Gřivna, Jakub Drápal, "Attacks on the confidentiality, integrity and availability of data and computer systems in the criminal case law of the Czech Republic", *Digital Investigation*, Volume 28, Mar. 2019, pp. 1-13.

[36] S. B. Alias, S. Manickam and M. M. Kadhum, "A Study on Packet Capture Mechanisms in Real Time Network Traffic," *2013 International Conference on Advanced Computer Science Applications and Technologies*, Kuching, 2013, pp. 456-460.

[37] P. Goyal and A. Goyal, "Comparative study of two most popular packet sniffing tools-Tcpdump and Wireshark," *2017 9th International Conference on Computational Intelligence and Communication Networks (CICN)*, Girne, 2017, pp. 77-81.

[38] S. Sandhya, S. Purkayastha, E. Joshua and A. Deep, "Assessment of website security by penetration testing using Wireshark," *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, 2017, pp. 1-4.

- [39] L. Bošnjak, J. Sreš and B. Brumen, "Brute-force and dictionary attack on hashed real-world passwords," *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, 2018, pp. 1161-1166.
- [40] R. R. Rohrmann, V. J. Ercolani and M. W. Patton, "Large scale port scanning through tor using parallel Nmap scans to scan large portions of the IPv4 range," *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, Beijing, 2017, pp. 185-187.
- [41] T. Churi, P. Sawardekar, A. Pardeshi and P. Vartak, "A secured methodology for anti-phishing," *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, Coimbatore, 2017, pp. 1-4.
- [42] Salami Attack at "<https://ajmaurya.wordpress.com/2014/03/27/what-is-a-salami-attack/>", Date Accessed Aug. 21, 2019.
- [43] Trust Exploitation Attack at "<https://www.orbit-computer-solutions.com/type-of-network-attack-trust-exploitation/>", Date Accessed Aug. 21, 2019.
- [44] B. Pingle, A. Mairaj and A. Y. Javaid, "Real-World Man-in-the-Middle (MITM) Attack Implementation Using Open Source Tools for Instructional Use," *2018 IEEE International Conference on Electro/Information Technology (EIT)*, Rochester, MI, 2018, pp. 0192-0197.
- [45] V. Radhakishan and S. Selvakumar, "Prevention of Man-in-the-Middle Attacks Using ID Based Signatures," *2011 Second International Conference on Networking and Distributed Computing*, Beijing, 2011, pp. 165-169.
- [46] E. Letsoalo and S. Ojo, "Session hijacking attacks in wireless networks: A review of existing mitigation techniques," *2017 IST-Africa Week Conference (IST-Africa)*, Windhoek, 2017, pp. 1-9.
- [47] Nithun Chand O and S. Mathivanan, "A survey on resource inflated Denial of Service attack defense mechanisms," *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, Coimbatore, 2016, pp. 1-4.
- [48] S. Suriadi, D. Stebila, A. Clark and H. Liu, "Defending Web Services against Denial of Service Attacks Using Client Puzzles," *2011 IEEE International Conference on Web Services*, Washington, DC, 2011, pp. 25-32.
- [49] K. S. Bhosale, M. Nenova and G. Iliev, "The distributed denial of service attacks (DDoS) prevention mechanisms on application layer," *2017 13th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS)*, Nis, 2017, pp. 136-139.
- [50] A. Degirmencioglu, H. T. Erdogan, M. A. Mizani and O. Yilmaz, "A classification approach for adaptive mitigation of SYN flood attacks: Preventing performance loss due to SYN flood attacks," *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, Istanbul, 2016, pp. 1109-1112.
- [51] W. Michiels, "Opportunities in White-Box Cryptography," in *IEEE Security & Privacy*, vol. 8, no. 1, pp. 64-67, Jan.-Feb. 2010.

- [52] V. Sagar and K. Kumar, "A symmetric key cryptography using genetic algorithm and error back propagation neural network," *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, 2015, pp. 1386-1391.
- [53] S. Chandra, S. Paira, S. S. Alam and G. Sanyal, "A comparative survey of Symmetric and Asymmetric Key Cryptography," *2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE)*, Hosur, 2014, pp. 83-93.
- [54] T. M. Sharma and R. Thilagavathy, "Performance analysis of advanced encryption standard for low power and area applications," *2013 IEEE Conference on Information & Communication Technologies*, Thuckalay, Tamil Nadu, India, 2013, pp. 967-972.
- [55] T. K. Jishamol and K. Rahimunnisa, "Low power and low area design for advanced encryption standard and fault detection scheme for secret communications," *2013 International Conference on Communication and Signal Processing*, Melmaruvathur, 2013, pp. 743-747.
- [56] N. Su, Y. Zhang and M. Li, "Research on Data Encryption Standard Based on AES Algorithm in Internet of Things Environment," *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Chengdu, China, 2019, pp. 2071-2075.
- [57] Z. Zainuddin and E. V. Manullang, "E-learning concept design of rijndael encryption process," *Proceedings of 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE)*, Bali, 2013, pp. 737-741.
- [58] M. B. Yassein, S. Aljawarneh, E. Qawasmeh, W. Mardini and Y. Khamayseh, "Comprehensive study of symmetric key and asymmetric key encryption algorithms," *2017 International Conference on Engineering and Technology (ICET)*, Antalya, 2017, pp. 1-7.
- [59] M. Biglari, E. Qasemi and B. Pourmohseni, "Maestro: A high performance AES encryption/decryption system," *The 17th CSI International Symposium on Computer Architecture & Digital Systems (CADS 2013)*, Tehran, 2013, pp. 145-148.
- [60] M. Noura, H. N. Noura, A. Chehab, M. M. Mansour and R. Couturier, "S-DES: An efficient & secure DES variant," *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*, Jounieh, 2018, pp. 1-6.
- [61] L. Jie, L. Yuxiang, S. Huafang and S. Weiwei, "A Power Analysis Resistant DES Cryptographic Algorithm and Its Hardware Design," *2012 Third International Conference on Digital Manufacturing & Automation*, GuiLin, 2012, pp. 121-124.
- [62] K. Mohamed, O. Mohamed, M. Hamoudi and M. Masmoudi, "QoS evaluation in VoIP software with and without Blowfish encryption module," *2015 3rd International Conference on Control, Engineering & Information Technology (CEIT)*, Tlemcen, 2015, pp. 1-6.
- [63] T. Nie and T. Zhang, "A study of DES and Blowfish encryption algorithm," *TENCON 2009 - 2009 IEEE Region 10 Conference*, Singapore, 2009, pp. 1-4.
- [64] A. Boicea, F. Radulescu, C. Truica and C. Costea, "Database Encryption Using Asymmetric Keys: A Case Study," *2017 21st International Conference on Control Systems and Computer Science (CSCS)*, Bucharest, 2017, pp. 317-323.

- [65] F. J. Aufa, Endroyono and A. Affandi, "Security System Analysis in Combination Method: RSA Encryption and Digital Signature Algorithm," *2018 4th International Conference on Science and Technology (ICST)*, Yogyakarta, 2018, pp. 1-5.
- [66] R. S. Dhakar, A. K. Gupta and P. Sharma, "Modified RSA Encryption Algorithm (MREA)," *2012 Second International Conference on Advanced Computing & Communication Technologies*, Rohtak, Haryana, 2012, pp. 426-429.
- [67] R. Y. Chang, S. Lin and W. Chung, "Diffie-Hellman key distribution in wireless multi-way relay networks," *2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, Kaohsiung, 2013, pp. 1-4.
- [68] Nan Li, "Research on Diffie-Hellman key exchange protocol," *2010 2nd International Conference on Computer Engineering and Technology*, Chengdu, 2010, pp. V4-634-V4-637.
- [69] B. Qing-Hai, Z. Wen-Bo, J. Peng and L. Xu, "Research on Design Principles of Elliptic Curve Public Key Cryptography and Its Implementation," *2012 International Conference on Computer Science and Service System*, Nanjing, 2012, pp. 1224-1227.
- [70] X. Fang and Y. Wu, "Investigation into the elliptic curve cryptography," *2017 3rd International Conference on Information Management (ICIM)*, Chengdu, 2017, pp. 412-415.
- [71] M. Janagan and M. Devanathan, "Area compactness architecture for elliptic curve cryptography," *International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME-2012)*, Salem, Tamilnadu, 2012, pp. 131-134.
- [72] Chen Tianhuang and Xu Xiaoguang, "Digital signature in the application of e-commerce security," *2010 International Conference on E-Health Networking Digital Ecosystems and Technologies (EDT)*, Shenzhen, 2010, pp. 366-369.
- [73] W. J. Buchanan, S. Li and R. Asif, "Lightweight cryptography methods", *Journal of Cyber Security Technology*, 23 Sep. 2017, pp. 187-201.
- [74] S. B. Sadkhan and A. O. Salman, "A survey on lightweight-cryptography status and future challenges," *2018 International Conference on Advance of Sustainable Engineering and its Application (ICASEA)*, Wasit, 2018, pp. 105-108.
- [75] I. Bhardwaj, A. Kumar and M. Bansal, "A review on lightweight cryptography algorithms for data security and authentication in IoTs," *2017 4th International Conference on Signal Processing, Computing and Control (ISPCC)*, Solan, 2017, pp. 504-509.
- [76] H. D. Zodpe, P. W. Wani and R. R. Mehta, "Design and implementation of algorithm for DES cryptanalysis," *2012 12th International Conference on Hybrid Intelligent Systems (HIS)*, Pune, 2012, pp. 278-282.
- [77] A. Pestunov, "Formalization and Systematization of Differential Cryptanalysis Notions," *2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Kitakyushu, 2014, pp. 542-545.
- [78] A. Bechtsoudis and N. Sklavos, "Side Channel Attacks Cryptanalysis against Block Ciphers Based on FPGA Devices," *2010 IEEE Computer Society Annual Symposium on VLSI*, Lixouri, Kefalonia, 2010, pp. 460-461.

- [79] T. Vateva-Gurova and N. Suri, "On the Detection of Side-Channel Attacks," *2018 IEEE 23rd Pacific Rim International Symposium on Dependable Computing (PRDC)*, Taipei, Taiwan, 2018, pp. 185-186.
- [80] M. Tang *et al.*, "Side-Channel Attacks in a Real Scenario," in *Tsinghua Science and Technology*, vol. 23, no. 5, pp. 586-598, Oct. 2018.
- [81] S. Jiao, T. Lei, Y. Gao, Z. Xie and X. Yuan, "Known-plaintext attack and ciphertext-only attack for encrypted single-pixel imaging," in *IEEE Access*, pp. 1-1, Aug. 2019.
- [82] Y. Pan and Y. Deng, "A Ciphertext-Only Attack Against the Cai-Cusick Lattice-Based Public-Key Cryptosystem," in *IEEE Transactions on Information Theory*, vol. 57, no. 3, pp. 1780-1785, March 2011.
- [83] Y. Zheng, M. Schulz, W. Lou, Y. T. Hou and M. Hollick, "Highly Efficient Known-Plaintext Attacks Against Orthogonal Blinding Based Physical Layer Security," in *IEEE Wireless Communications Letters*, vol. 4, no. 1, pp. 34-37, Feb. 2015.
- [84] Yin Yan-Qing and Long Zhuo-Min, "Chosen plaintext attacking on a hyper-chaos image encryption algorithm," *2013 IEEE 4th International Conference on Electronics Information and Emergency Communication*, Beijing, 2013, pp. 144-147.
- [85] A. Pestunov, "Formalization and Systematization of Differential Cryptanalysis Notions," *2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Kitakyushu, 2014, pp. 542-545.
- [86] K. Zhang, J. Guan, B. Hu and D. Lin, "Integral cryptanalysis on Simeck," *2016 Sixth International Conference on Information Science and Technology (ICIST)*, Dalian, 2016, pp. 216-222.
- [87] M. Kumagai, Y. Musashi, D. A. L. Romana, K. Takemori, S. Kubota and K. Sugitani, "SSH Dictionary Attack and DNS Reverse Resolution Traffic in Campus Network," *2010 Third International Conference on Intelligent Networks and Intelligent Systems*, Shenyang, 2010, pp. 645-648.
- [88] J. Kokes and R. Lorencz, "Linear cryptanalysis of Baby Rijndael," *2015 Forth International Conference on e-Technologies and Networks for Development (ICeND)*, Lodz, 2015, pp. 1-6.
- [89] M. Rodinko and R. Oliynykov, "Open problems of proving security of ARX-based ciphers to differential cryptanalysis," *2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, Kharkov, 2017, pp. 228-231.
- [90] Xu Dewu and Chen Wei, "A survey on cryptanalysis of block ciphers," *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, Taiyuan, 2010, pp. V8-218-V8-220.
- [91] Cryptanalyst tools at "<https://resources.infosecinstitute.com/cryptanalysis-tools/#gref>", Date Accessed Aug. 26, 2019.
- [92] Distributed Network Attack at "<https://www.tracip.fr/cryptographie-dechiffrement/distributed-network-attack.html>", Date Accessed Aug. 27, 2019.

- [93] Distributed Network Attack with Frontier at “<https://www.parabon.com/case-studies/distributed-network-attack-enterprise.html>”, Date Accessed Aug. 27, 2019.
- [94] Distributed Network Attack with Frontier File & Password Decryption at “<https://www.parabon.com/frontier-powered-apps/distributed-network-attack-enterprise.html>”, Date Accessed Aug. 27, 2019.
- [95] AccessData at “<https://accessdata.com/>”, Date Accessed Aug. 27, 2019.
- [96] Parabon’s Frontier Grid Platform at “<https://www.parabon.com/frontier-grid-platform/>”, Date Accessed Aug. 27, 2019.
- [97] S. J. Aboud, "An efficient method for attack RSA scheme," *2009 Second International Conference on the Applications of Digital Information and Web Technologies*, London, 2009, pp. 587-591.
- [98] N. Floissac and Y. L'Hyver, "From AES-128 to AES-192 and AES-256, How to Adapt Differential Fault Analysis Attacks on Key Expansion," *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Nara, 2011, pp. 43-53.
- [99] R. Yang, L. Wallace and I. Burchett, “Teaching Cryptology At All Levels Using CrypTool”, *15th Colloquium for Information Systems Security Education*, Fairborn, Ohio, Jun. 13-15, 2011.
- [100] M. Denis, C. Zena and T. Hayajneh, "Penetration testing: Concepts, attack methods, and defense strategies," *2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, Farmingdale, NY, 2016, pp. 1-6.
- [101] Heartbleed Vulnerability at “<https://hakin9.org/detecting-and-exploiting-the-openssl-heartbleed-vulnerability/>”, Date Accessed Sep. 10, 2019.
- [102] RSA Encryption and Decryption Formulas at “<https://www.comparitech.com/blog/information-security/rsa-encryption/>”, Date Accessed Sep. 11, 2019.
- [103] RSA 768-bit Key broken at “https://www.theregister.co.uk/2010/01/07/rsa_768_broken/”, Date Accessed Sep. 11, 2019.
- [104] S. Cavallar et al. “Factorization of a 512-bit RSA Modulus”, *Citibank, 1 North Gate Road, Mendham, NJ 07945–3104, USA*.