History, Theory, and Serial Optimization of the Advanced Encryption System

by
Liam J. Cates

A Senior Project
presented to the Faculty of the Computer Science Department
College of Engineering
California Polytechnic State University, San Luis Obispo

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science

supervised by
Dr. Chris Lupo
Computer Science Department Chair

September, 2019

Dedicated to the

Katz Family

for their support, mentorship, and love

Table of Contents

Figures

Tables

Introduction

This report guides the unacquainted reader to develop an understanding of the context, foundations, and mechanism implemented by the world's most widely used cryptographic system, the Advanced Encryption Standard (AES). There is quite a lot of ground to cover and we shall attempt it with a high degree of efficiency. What follows is an overview of the 3 parts that comprise this report along with constituent sections.

Part 1, A History of Symmetric Key Block Ciphers, provides a functional understanding of the fundamental properties, background, and historical context of the AES.

Section 1, Binary Data, focuses on the format of the inputs and outputs utilized by all digital processes, including those of the AES. The section introduces the process of data expression via binary numeral systems, the conception of this methodology, its initial applications, contemporary definitions and standards, modern representation, and then transitions into the topic of data security by cryptography.

Section 2, Modernization of Cryptography, explains two concepts that propelled cryptography from a secretive art to a modern science: the shift away from security by obscurity and system security via the use of cryptographic keys. We then move on to a summary of modern cryptographic study, applications, and techniques to arrive at a breakdown of the first ever encryption method proven to be perfectly secure, the One Time Pad. We explore the practical implications, and the weaknesses of this perfectly secure system.

Section 3, Symmetric Key Cryptography, addresses the modern derivatives of the One Time Pad with great attention to Symmetric Key Block Ciphers. We explore the impact of the first national cryptographic standard, some relevant definitions, the principles of symmetric cipher design, and the issues of key management. Finally we become acquainted with the AES, of which, the mathematical foundations and an example implementation are the focus of the remainder of this report.

Part 2, Mathematics of the Advanced Encryption Standard, addresses the mathematical concepts necessary for a foundational understanding of AES operations.

Section 4, Set Theory, examines the elements operated upon by the cryptographic primitives of the AES. These systems are implemented with finite elements. Sets with which most are familiar, such as the set of integers, are infinite. We work with a functional definition of sets, known as naive set theory, to define unordered collections of distinct objects. A given Set is defined by the properties of its members, whether by shared properties, or a lack thereof, such as the set of even numbers, which share evenness, and lack oddness. Once a definition is established, we are able to analyse a Set and determine the algebraic properties held by relations between set members, represented by mathematical operations of varied complexity.

Section 5, Number Theory, facilitates the understanding of cryptographic algorithms through the examination of relevant mathematical operations. Modern cryptographic primitives implemented in both symmetric and asymmetric ciphers are based on arithmetic within a finite number of elements. Not only is modular arithmetic a common way of performing arithmetic in a finite set of integers, it is the method implemented by the AES. As such, understanding modular arithmetic and its application is of fundamental importance in the context of this report as well as in the greater scope of modern cryptographic study and practice.

Section 6, Abstract Algebra, concerns those properties that define the algebraic structures which mathematically model the mechanisms of the AES. Through selective abstraction, mathematicians have defined algebraic structures now integral to both pure mathematics and the applied sciences. We shall see how the definition of a binary operation and the relation it creates, affects the set for which it is defined. We begin from the most basic structure, a general set for which a binary operation is defined, then progress through the individual properties necessitated by the operations of the AES and define the algebraic structures which result as they increase in complexity. This incremental addition of properties creates a hierarchy of algebraic structures. Groups, rings, and fields constitute the basic hierarchy of abstract algebraic objects and are required for the definition and understanding of the AES.

Section 7, Galois Field Arithmetic, enumerates the arithmetic operations defined for the AES. The transformations implemented by the AES operate within the Galois Field of 256 members known as $GF(2^8)$. Within this algebraic structure we redefine the arithmetic operations of addition, subtraction, multiplication, and division such that these operations, when performed on the underlying set, remain consistent with the behaviour expected of an infinite set. When a system such as this has been properly defined, it allows us to perform finite field, or Galois Field, arithmetic, that is, perform operations with finite members which adhere to arithmetic laws consistent with a field of finite members. Due to the property of closure, maintained by the algebraic structure of this field, we are assured that all mathematical operations defined in the field result in an 8-bit number. Thus operands and resultants are limited to the range $0 \leq i < 2^8$, represented by numbers from 0 to 255. Rigourous definition of the mathematics involved grants an assurance of security, and the ability to keep intermediate results under 8-bits creates substantial increases in efficiency.

Part 3, Specification of the Advanced Encryption Standard, defines an algorithmic specification for the AES which includes it's notation, inputs, and outputs as well as conventions for describing them, a full specification of the AES system which provides message confidentiality for a cipher block under a cryptographic key, and modes of operation, which extends block cipher security to the whole of a message.

Section 8, Notation and Conventions, defines the notation and conventions used to model and algorithmically specify the AES. This includes the ordering and indexing of bits, bytes, and words which comprise the AES operands, as well as descriptions of the parameters and resultants that characterize the key expansion, encryption, and decryption routines of the AES.

Section 9, Functions, details a method by which the mechanisms of the AES' cryptographic transformations maybe algorithmically implemented. As a key iterated product cipher, the AES executes a number of round function iterations, on a block of plaintext bits, as discussed in Section 3.3. Each round transformation is executed in the same manner, with variance provided by round values, generally called round constants and a round key. The AES is fundamentally composed of a key schedule and a block cipher. The AES key schedule algorithm calculates round keys and the AES cipher consists of the round function, specifically composed of four byte-oriented transformations

Section 10, Block Cipher Modes of Operation, explores a few basic standardised block cipher modes of operation, these procedures were developed to extend block cipher capability. Individually, a block cipher serves to provide message confidentiality, protection from unauthorized access. However, a block cipher is only defined for a single block size transformation per key. In practice, the size of a message is larger than the block size, often much larger. It is because of this that block ciphers are classified as cryptographic primitives, to be used as a component in a secure cryptosystem. While the applications of block cipher modes are many, the few focused by this document are procedures that allow a generic block cipher to transform data allocations larger than a single block and achieve secure results under a fixed key.

Part 1

A Brief History of the Advanced Encryption Standard

Section 1 : Binary Data

1.1 : The Binary Numeral System

All data is represented as sequences of symbols of a finite or well-defined set. (Greenlaw & Hoover 1998) While data can be represented by any system of variance, the most abundant is the contemporary binary system used by digital devices. (Greenlaw & Hoover 1998) A digital device represents a singular piece of data by sequences drawn from a "binary alphabet", a term that defines a set that contains two distinct entities. (Greenlaw & Hoover 1998) Any number can be represented by a sequence of binary digits. (Greenlaw & Hoover 1998) Theoretically, a binary value represents a sequence of "ON" and "OFF". (Greenlaw & Hoover 1998) Functionally, a binary 0 indicates off while a binary 1 indicates an electrical signal or base 2 exponent that is turned on. (Greenlaw & Hoover 1998) Table 1, below, illustrates this system using the binary value 10100110:

Exponent:	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Value:	128	64	32	16	8	4	2	1
ON/OFF:	1	0	1	0	0	1	1	0

Table 1

To derive the value of a given binary representation in Table 1, we place the binary digits in a column. Each binary digit is then aligned with a sequence of exponentiation of the number, the base, two, with the exponent's value beginning at 0 increasing by one as we move digit positions from right to left. We interpret the value of the binary sequence by summing "ON" column values, those that contain a 1 to achieve: $(128 + 32 + 4 + 2) = 166$. Thusly, any mechanism capable of two mutually exclusive states may represent a given binary value. By example, the following binary sequences shown in Figure 1, below, are interpretable as the value 166:

1	0	1	0	0	1	1	0
■	□	■	□	□	■	■	□
y	n	y	n	n	y	y	n
+	-	+	-	-	+	+	-

Figure 1

Data is measured by the bit, a basic unit of information in the field of Information Theory.(Shannon 1948) The bit is an abstraction of a logical value possessing one of two distinct states.(Shannon 1948)

Examples of such a system are abundant, even in nature, day and night, magnetic and electric polarity, or the state of a neuron. In the realm of information theory, a bit is alternatively called a *shannon (Sh)*.[Rowlett 2018] Claude Shannon was responsible for both developing the field of information theory and publishing "*bit"* in 1948.[Shannon 1948] The definition of a shannon is derived from the central tenet of information theory, entropy, the general concept that unlikely events carry more information. "...if a volcano rarely erupts, then a message that it is erupting is more informative than a message it is not erupting".[Rowlett 2018 par shannon] By definition, a message of probability p has an information content equal to $log_2 p$ shannons.[Rowlett 2018] As an example, if the set of data symbols consists only of the 26 lowercase letters of the English alphabet, with all strings being equally likely, then the probability of a message of length 10 is $(1/26^{10})$ and its information content is $10(log_2 26) = 47.004\ Sh$. A single shannon represents the information content of an event with probability 1⁄2, $1(log_2 2) = 1\ Sh$ [Rowlett 2018] This means a single shannon defines the outcomes of a system where there is equal probability of either of two outcomes, a theoretical coin toss, which we can represent with one bit. By extension, a bit sequence of a given number, with all possible bit sequences being equally likely, has an information content, expressed in shannons, equal to the number of bits in the sequence.[Rowlett 2018] For this, the unit was originally the bit.[Rowlett 2018] Shannon credits the coinage of the "bit" unit to John W. Tukey, who also coined "software".[Buchholz 2000 p. 69] Tukey uses "bit" as a portmanteau for "binary digit".[Shannon 1948 p. 1]

A bit is physically represented by a two-state device; any mechanism that exists and may switch between one of two possible states.[Shannon 1948] Two state devices have a myriad of physical implementations in modern digital devices, as relative levels of charge or voltage, a sequence of current pulses, or the state of a circuit.,[Kuphaldt 2001), (AAKCT 2018] A majority of modern digital devices use positive logic, the expression of a logical or digital value of 1 by a more positive value relative to the representation of 0.[Kuphaldt 2001] Dynamic random-access memory uses semiconductors to represent logical binary values via capacitors with two levels of electric charge.[AAKCT 2018] Compact discs, Blu-ray, and other optical disc technologies encode data via microscopic indentations on a reflective surface, called pits.[BDA 2010] One dimensional bar codes use the thickness of parallel, alternating black and white lines to encode binary values.[Woodland & Bernard 1949] Familiar symbology, such as alphabetic letters, numerical digits, punctuation marks, and others rely on popular character encoding conventions to translate from the state of data stored on a physical device to human interpretable symbols.[Greenlaw & Hoover 1998]

1.2 : Conception of the Binary Numeral System

Our contemporary binary encoding scheme was conceptualized in Europe during the 17th century. Thomas Harriot, an English mathematician of incredible accomplishment[(Apt 2019)], recorded the first use of a binary numeral system c. 1600 AD.[(O'Connor & Robertson 2019)] In fact, Harriot had, "considered working with not only binary systems, but ternary, quaternary, quinternary, and higher systems as well". [(O'Connor & Robertson 2019 par. 3)] Harriot's studies were often secretive, his results unpublished for fear of being labeled a heretic. Records were found after his death, among 7000 pages of notes.[(Apt 2019)] These notes, now housed by the British Museum, contain a page with the information displayed by Table 2, to the right. [(Shirley 1951)]

1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000

Table 2

In 1605, Francis Bacon, credited with inventing the scientific method[(Ochulor 2011)], developed an encoding by which the Latin alphabet could be reduced to binary sequences,[(Gallup 2010)] a logical predecessor of modern encoding. Bacon's cypher, a substitution cipher, is given by Table 3, below.[(Gallup 2010)] Developing a general theory of binary encoding Bacon deduced these digits could be represented by any abstraction, "provided those objects be capable of a twofold difference only", offering examples such as, "by Bells, by Trumpets, by Lights and Torches, by the report of Muskets, and any instruments of like nature".[(Gallup 2010 p. 67)]

A	aaaaa	G	aabba	N	abbaa	T	baaba
B	aaaab	H	aabbb	O	abbab	U & V	baabb
C	aaaba	I & J	abaaa	P	abbba	W	babaa
D	aaabb	K	abaab	Q	abbbb	X	babab
E	aabaa	L	ababa	R	baaaa	Y	babba
F	aabab	M	ababb	S	baaab	Z	babbb

Table 3

The first known publication of a binary system was by Juan Caramuel y Lobkowitz (JCL) a prodigious Spanish scholar and author.[(O'Connor & Robertson 2010)] In 1670 JCL published an encyclopaedia of mathematics entitled Mathesis Biceps: Vetus Et Nova. This work defined the general principle and outlined benefits of numeric systems other than the predominant "base 10". Donald Knuth, noted American computer scientist wrote of JCL's work in The Art of

Computer Programming "Caramuel discusses the representation of numbers in radices 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, and 60 at some length, but gave no examples of arithmetic operations in nondecimal systems (except for the trivial operation of adding unity)." (O'Connor & Robertson 2010 par 6)

While publication of the modern binary system can be traced back to recent centuries, systems related to binary numbers have evolved throughout various ancient cultures. Gottfried Leibniz, a distinguished German polymath and philosopher, was specifically inspired by an ancient Chinese divination text, the I Ching.(Smith 2008) Leibniz's fascination with binary numerals was inspired by his theology.(Smith 2008) Leibniz interpreted depictions on the I Ching as corresponding to 64 binary representations, from 0 to 111111.(Wilhelm & Baynes 1967) Leibniz believed this to be evidence of the universality of binary arithmetic(Smith 2008) and symbolic of the Christian Creationist theory "*creatio ex nihilo"* or "creation out of nothing".(Smith 2008 p. 450)

However, Leibniz, binary's self-proclaimed inventor, may have plagiarized. He was familiar with the works of both Juan Caramuel y Lobkowitz and Thomas Harriot.(ALLM 2018) Thomas Harriot demonstrated representation by a base 2 system(O'Connor & Robertson 2019), while Juan Caramuel y Lobkowitz worked with a variety of base values and their logarithms including base 2.(O'Connor & Robertson 2010) Regardless, Leibniz vastly expanded both binary and formal logic through the discovery of properties such as "conjunction, disjunction, negation, identity, inclusion, and the empty set".(Lande 2014 p. 21) As with the present system of binary, Leibniz's used 0 and 1, as is displayed by Table 4, below. This work appears in *Explication de l'Arithmétique Binaire,* 1703.(Strickland 2007)

2^0	0 0 0 1
2^1	0 0 1 0
2^2	0 1 0 0
2^3	1 0 0 0

Table 4

Additionally, in 1679, during his study of binary arithmetic, Leibniz imagined a machine that represented binary numbers by marbles, governed by a positional system of open and closed gates which the marbles gravity.(Lande 2014) "Modern electronic digital computers have replaced Leibniz's gravity-driven marbles with shift registers, voltage gradients, and electron pulses, but otherwise they run roughly as Leibniz had visualized"(Agarwal & Sen 2014 p28)

1.3 : Popularization of Binary Data

The first known application of a physical binary representation of data was by Basile Bouchon in 1725.(Heudin 2008) Bouchon developed a punched paper tape that allowed semi-automated set-up of a textile loom.(Heudin 2008) It was not until 1804, after further developments by Jean Baptiste Falcon in 1728(Essinger 2004) and Jacques de Vaucanson in 1740s(Essinger 2004), that this system saw its first commercial success. The first widespread adoption of binary was through the use of the Jacquard Loom.(Essinger 2004) Its inventor, Joseph Marie Jacquard, used punched cards joined to form a loop such that individual portions of the input length could be edited.(Essinger 2004) The term "Jacquard" refers to a modular control mechanism responsible for automating the weaving of complex patterns, rather than a specific loom type or design.(DMS 2017) The Jacquard loom was an international success and punch cards would go on to become an integral part of industry, research, and government data storage for the next two centuries.(Essinger 2004)

These types of physical mediums, paper tapes and cards, theoretically represent information as a sequence of positions, each position has either been perforated or not.(Essinger 2004) This system thus represents series of two-state devices and is able to represent one bit of information at each position. There is evidence of productions that used input card counts in the hundreds of thousands, including the first digitally manufactured book, with the physical medium woven by these looms rather than printed with ink.(Norman 2019) However, the most influential artifact, perhaps, was a portrait of Jacquard, woven in silk, owned by Charles Babbage. Made to order in 1839, these masterworks required 24000 punched cards and were crafted with a precision that Babbage deeply admired “sheet of woven silk, framed and glazed, but looking so perfectly like an engraving, that it had been mistaken for such by two members of the Royal Academy”.(Gross. 2015 par. 3) Regarded as a "father of the computer",(Halacy 1970) Babbage is considered to have both, invented the first mechanical computer,(Copeland 2017) as well as designed the first programmable computer.(Copeland 2017) The design of this programmable computer, the Analytic Engine, though conceived in 1837, was "essentially the same as that which has dominated computer design in the electronic era".(Swad 2019 par. 9) It was Jacquard's invention which inspired Babbage in using punch cards to store the binary data interpreted by the Analytic Engine.(Gross. 2015)

Nearly 50 years later, punched hole binary representation was adopted by the founder of the Tabulating Machine Company, Herman Hollerith.(Essinger 2004) In the 1880s, Hollerith was inspired by hole-punched railway tickets which represented categorical passenger data by which conductors, "... verified that the passenger occupying the seat was in fact the same who had originally presented the ticket".(CPRR.org 2014 par. 1) Hollerith developed a method of data storage on punched cards, readable by an automated mechanical process.(Da Cruz 2001) This application was revolutionary as all previous automated systems processed instruction lists, such as the

complex procedures which directed silk spinning Jacquard looms, and not arbitrary data of the sort Hollerith had in mind. "After some initial trials with paper tape, he [Hollerith] settled on punched cards…"[(Da Cruz 2001 par. 2)] An array of spring-loaded metal pins was positioned over a card ready for processing. A pattern of punched holes would allow a configuration of pins to "...pass through the holes, making contact with little wells of mercury, completing an electrical circuit"[(Da Cruz 2019 par. 1)] that was used to count or sort punched cards as well as ring a bell signaling a human operator that the current card had been processed and that the next needed to be hand fed.[(Da Cruz 2019)]

"Herman Hollerith was an American inventor and entrepreneur whose inventions paved the way for the information processing industry"[(Satyasikha. 2014 par. 1)] Specifically, Hollerith's tabulating machine marks a new beginning, as it was the first information processing system to successfully replace pen and paper.[(Da Cruz 2001)] Hollerith's electromechanical tabulators proved their worth by processing the data generated during the 1890 United States Census far faster than the 1880 census. The tabulating machines reduced what had been a ten-year job to three months, ultimately reducing 1890 taxpayers costs by five million dollars.[(Da Cruz 2001)] Hollerith's company would eventually become the core of IBM, producer of the dominant mainframe computer family, the System/360, an industry standard for the computing market during the 1960s and 1970s.[(Rosenbaum 1998)] IBM, "a direct descendant of the work that went on in Jacquard's workshop",[(Essinger 2004 p192-93)] forwarded the ubiquity of binary data representation and processing in both industry and government. While punched card data storage is generally obsolete, reports show that as recently as 2012 there were still .02% of active US voting machines using this method to record voter input as opposed to the 95% that used electronic voting machines and optically scanned paper ballots.[(ProCon 2013)]

1.4 : The Definitions and Standards for Collections of Binary Data

Bit collections are most commonly expressed via their byte length, a unit coined by Werner Buchholz in 1956.(Buchholz 1956) The byte is defined to represent the arbitrary length of a bit sequence used to encode a single textual character.(Buchholz 1962) Modern convention was established by the preeminence of IBM's System 360 computer line in the 1960's which utilized an eight bit byte.(Swad 2019) Due to variance in system design, "octet" explicitly defines an eight bit unit.(Bemer 2000) Bytes are relatively small data collections, modern computers manipulate "words".(Buchholz 1962) "A word consists of the number of data bits transmitted in parallel from or to memory in one memory cycle".(Buchholz 1962 p. 40) Thus, the system's structural properties define its word size.(Buchholz 1962) Historically, system designs have had their word's data length range from 1(Koblentz 2004) to 128(Waterman & Asanovi´c 2017) bits with a majority of modern systems featuring lengths of 32 or 64.(Buchholz 1962)

Collections of significant magnitude are most commonly quantified by their decimal multiples as defined by the International System of Units (SI) using prefixes appended to a given unit of measure.(NIST 2019) The SI units are the most widely used system of measure and the international standard for measurement.(NIST 2019) The SI prefixes are a defined series of decimal multiples of standardized units that includes the prefixes kilo (10^3) through yotta (10^{24}) increment by multiples of 1000 (10^3).(NIST 2019) 1000 bytes would be expressed as an SI unit if prefixed by k- as kbytes to mean 10^3 bytes or 1 kilobyte.(NIST 2019) The nature of the SI system is well suited to count, by powers of ten, those physical quantities for which we have long used a base 10 system. However, the logical quantities used in computing are represented by a binary system which is inherently base 2. Thus, for the purposes of computing, the SI prefixes were misappropriated for nearby binary multiples, eg $k = 2^{10}$.(McCullagh 2007) In the early years, there was no significant difference in using SI prefixes for either binary or decimal multiples.(McCullagh 2007) $2^{10} = 1024$ and $10^3 = 1000$, for example, are equal to two significant figures.(McCullagh 2007) As computational capacities increased the absolute error between the two interpretations rose causing issues for manufacturers and consumers alike. Ultimately, disparate prefix interpretation culminated in significant class action lawsuits.(McCullagh 2007)

A set of binary prefixes created to solve these issues was standardized by the International Electrotechnical Commission (IEC). The IEC is "the world's leading organization for the preparation and publication of International Standards for all electrical, electronic and related technologies".(IEC-W 2019 par. 2) IEC members adopt these publications as national standards. Member and affiliate countries influenced by IEC standards compose "more than 97% of the world's population".(IEC-W 2019 par. 2) In 1996(IUCr 1997), the IEC formulated its prefixes by contraction

between the first two letters of the popular SI prefixes and "bi" from *binary.*[IEC 2005] This results in the prefixes *kibi*, *mebi*, *gibi* and *tebi*, for which the corresponding symbols Ki, Mi, Gi and Ti were used.[Abrahams 2000] IEC employed the same contraction system to define *pebi* (Pi) and *exbi-* (Ei) via 60027-2 Amendment 2 (1999)[IEC 2005] as well as zebi- (Zi) and yobi- (Yi) via the third edition of IEC 60027 in 2005, thus ascribing a binary equivalent to all SI prefixes. A juxtaposition of the SI and IEC Prefixes for multiples of bits is displayed by Table 5, below.

Decimal				Binary			
Value		IS		Value		IEC	
1000	10^3	kilobit	kbit	1024	2^{10}	kibibit	Kibit
1000^2	10^6	megabit	Mbit	1024^2	2^{20}	mebibit	Mibit
1000^3	10^9	gigabit	Gbit	1024^3	2^{30}	gibibit	Gibit
1000^4	10^{12}	terabit	Tbit	1024^4	2^{40}	tebibit	Tibit
1000^5	10^{15}	petabit	Pbit	1024^5	2^{50}	pebibit	Pibit
1000^6	10^{18}	exabit	Ebit	1024^6	2^{60}	exbibit	Eibit
1000^7	10^{21}	zettabit	Zbit	1024^7	2^{70}	zebibit	Zibit
1000^8	10^{24}	yottabit	Ybit	1024^8	2^{80}	yobibit	Yibit

Table 5

The United States National Institute of Standards and Technology (NIST) supports the ISO/IEC "Prefixes for binary multiples" standards and hosts a website documenting their usage.[TMNT 1998] NIST recommends "in English, the first syllable of the name of the binary-multiple prefix should be pronounced in the same way as the first syllable of the name of the corresponding SI prefix, and that the second syllable should be pronounced as "bee"."[TMNT 1998 par. 2] NIST has determined that SI prefixes "refer strictly to powers of 10" and "should not be used to indicate powers of 2".[Thompson & Taylor 2008 p. 74] In summary, the symbol for the binary digit is either *bit* per recommendation by IEC 60027 and its successors, or the lowercase *b* symbol recommended by the IEEE 1541-2002.[IEEE 2009] Both standards recommend the use of (*o*) for octet and (B) for byte. A direct comparison of the SI and IEC Prefixes for multiples of bits is displayed by Table 6, below.

IEC prefix		Representations				Customary prefix	
Name	Symbol	Base 2	Base 1024	Value	Base 10	Name	Symbol
kibi	Ki	2^{10}	1024^{1}	1024	= 1.024×10^{3}	kilo	k or K
mebi	Mi	2^{20}	1024^{2}	1048576	≈ 1.049×10^{6}	mega	M
gibi	Gi	2^{30}	1024^{3}	1073741824	≈ 1.074×10^{9}	giga	G
tebi	Ti	2^{40}	1024^{4}	1099511627776	≈ 1.100×10^{12}	tera	T
pebi	Pi	2^{50}	1024^{5}	1125899906842624	≈ 1.126×10^{15}	peta	P
exbi	Ei	2^{60}	1024^{6}	1152921504606846976	≈ 1.153×10^{18}	exa	E
zebi	Zi	2^{70}	1024^{7}	1180591620717411303424	≈ 1.181×10^{21}	zetta	Z
yobi	Yi	2^{80}	1024^{8}	1208925819614629174706176	≈ 1.209×10^{24}	yotta	Y

Table 6

1.5 : Modern Data and Its Security

Less than 1% of the worlds storage capacity was digital in 1986, this grew to 94% by 2007.(Leontiou 2011) The year 2002 was when humanity utilized a greater margin of digital storage, an event considered to reflect our transition into the digital age.(Leontiou 2011) Every communication device uses the propagation of a signal to convey the information necessitated by its function.(Samson 1999) The term signal abstractly references "any kind of physical quantity that conveys information"(Kuphaldt 2001 (1) par. 2) . Examples of signals include the varying voltage, current, and electromagnetic waves used in modern digital technology.(Kuphaldt 2001) There are two types of signals, digital and analog.(Kuphaldt 2001) While both signals propagate through the same mediums, it is their interpretation that differs.(Kuphaldt 2001) A device that interprets analog signals does so as a real number in a range of continuous values.(Kuphaldt 2001)

Conversely, digital devices interpret a sequence of discrete values from bands of signal level.(Samson 1999) All signal values within a band represent one of a set of predetermined information states.(Samson 1999) "...one of the first electronic digital computers, the Eniac. The designers of the Eniac chose to represent numbers in decimal form, digitally… …This approach turned out to be counter-productive, and virtually all digital computers since then have been purely binary in design."(Kuphaldt 2001 (2) par. 4) A binary signal is a digital signal capable of only two values, derived from the separation of the continuous signal range into discrete bands, a process generally known as discretization.(Samson 1999) These values correspond to binary 1 or 0, and a signal is then modulated, falling into either one of the discrete bands, to represent the value of a bit.(Samson 1999) A diagrammatic representation of binary signal discretization is shown by Figure 2, below.

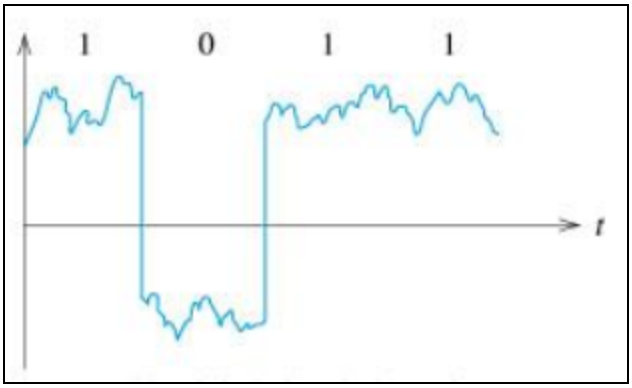

Figure 2

Inherent in any sort of signal propagation is the risk of inaccurate data transfer due to system noise. System noise is signal interference, this interference manifests as changes in the interpretation of signal level.(Kuphaldt 2001 (3)) While noise always degrades the quality of an analog signal, because of the high discretization associated with a binary signal, noise of reasonable amplitude does not leave its intended signal band, and has no effect on the signal value interpreted.(Kuphaldt 2001 (3))

Transmission of a data signal, the communication of information from one location to another, requires a communication "channel". The term channel is a generalization for some arbitrary signal pathway or medium of transmission.(Stallings 2007) Transmission is of two types, guided transmission via physical mediums (i.e. twisted-pair wire, cable, and fiber-optic cable) and unguided transmission via broadcast mediums (i.e. microwave, satellite, radio, and infrared).(Stallings 2007) Our digital information travels across vast networks owned by various corporations, is hosted by countries around the world, stored on complex devices accessed by an indefinite number of users, and will persist long after we ourselves have gone. Vast resources are required to create a secure communication channel as the entire span would require constant physical protection.(Van Tilborg & Jajodia 2011) Thus, the majority of modern communication and exchange of private data, is over an insecure channel. A need for secure communication indicates the existence of an unwanted presence, called an adversary or attacker.(Van Tilborg & Jajodia 2011) Such presence is classified as either a passive attacker, such as an eavesdropper, or an active attacker, such as a cryptanalyst.(Van Tilborg & Jajodia 2011)

These adversaries perform actions, such as eavesdropping, defined as "the surreptitious monitoring of communication".(Van Tilborg & Jajodia 2011 p. 378) To defeat adversaries, such as eavesdroppers, a signal's sender may apply countermeasures to protect the contents of communication. The most common countermeasures come from the field of Cryptology.(Van Tilborg & Jajodia 2011) "Cryptology is the discipline of cryptography and cryptanalysis and of their interaction".(Van Tilborg & Jajodia 2011 p. 283) The field of modern cryptologic application consists of the study and practical application of systems which overcome adversaries seeking to compromise secure communication.(Menezes Van Oorshot & Vanstone 1997) The necessity of cryptography arises when the security of information is required until authorized retrieval. This includes communications between individuals separated by space and storage between instances in time.(Van Tilborg & Jajodia 2011)

Suppose Alice and Bob wish to share information. For this they can rely on secure communication via cryptographic protocol, a distributed algorithm describing a procedure of exchange for two or more parties that achieves certain security objectives.(Van Tilborg & Jajodia 2011) Interaction via cryptographic protocol is done through the exchange of data messages.(Van Tilborg & Jajodia 2011) We will use the term message as a general identifier for information to be transmitted. Bank statements, medical test results, and intimate letters are among the messages Alice and

Bob might share privately. In the most basic case, an encryption operation performed by the sender, and a decryption operation performed by the receiver is necessary for message confidentiality.(Van Tilborg & Jajodia 2011) Cryptographically, decryption is the inverse of encryption.(Van Tilborg & Jajodia 2011) Alice encrypts a private message m, using an encryption algorithm $E()$, yielding the ciphertext c, $E(m) = c$.(Van Tilborg & Jajodia 2011) Bob receives the ciphertext, supplies it to a corresponding decryption algorithm $D()$, and restores the message m, $D(c) = m$, also known as the plaintext.(Van Tilborg & Jajodia 2011) The advantage of this complexity can be seen in the modern applications of cryptography, the object of which is secure information transfer and storage. When using a well designed and properly implemented cryptographic algorithm, an intercepted ciphertext will contain no information to differentiate the contents of that ciphertext from truly random bits.(Van Tilborg & Jajodia 2011)

Section 2 : Modernization of Cryptography

2.1 : The End of Security by Obscurity

Cryptography was first applied by the Egyptians around 4000 years ago.(Menezes Van Oorshot & Vanstone 1997) As they developed, cryptographic methods were implemented to ensure secrecy of critical communications. Its predominant practitioners were associated with government in general, whether military or diplomatic, e.g. spies, diplomats, military officers, and heads of state.(Menezes Van Oorshot & Vanstone 1997) Prior to the modern age, Cryptographic application primarily consisted of encryption. Encryption is a process that provides message confidentiality.(Paar & Pelzi 2009) The confidentiality of a message is generally achieved by mechanisms which preclude access to information.(Paar & Pelzi 2009) Encryption is a method of message conversion capable of transforming information from a comprehensible form into an incomprehensible one, and back, when necessary.(Van Tilborg & Jajodia 2011) These techniques encode messages in such a way that only parties with secret knowledge are allowed access.

In security engineering, a dependance on secrecy of design or implementation as the primary security mechanism is known as security through, or by, obscurity.(Paar & Pelzi 2009) The first security expert to bring the faults of this methodology to public attention was locksmith Alfred Charles Hobbs. In an 1851 demonstration, Hobbs displayed how even the most advanced locks were defeated by common criminals. The public reaction assumed that exposing such design faults would make these systems more susceptible to attack. Hobbs replied "Rogues are very keen in their profession, and know already much more than we can teach them".(Stross 2006 par. 25) Today, security by obscurity as a system's only security mechanism is discouraged by standards bodies. The United States National Institute of Standards and Technology (NIST) states, "System security should not depend on the secrecy of the implementation or its components."(Scarfone Jansen Tracy 2008 p. 2-4)

The practices of security by obscurity oppose the modern practices of security by design principles and open design.(Stallings 2017) A practical example of the success of these latter two methodologies is the open source operating system Linux. Linux, arguably the most famous example of open-source software, has never had the opportunity to use secrecy as a source of security.(Germain 2016) Moreover, the fact that the Linux source code is widely available improves the odds that any flaws will be found sooner and solved more efficiently, a phenomenon known as Linus's Law.(Raymond 2000) Today, Linux is hardy, secure, and robust, with "the largest installed base of all general-purpose operating systems".(Germain 2016 par. 5) In contrast, keeping the specification of a widely used method classified is a near impossibility. Individual adversaries might bribe, blackmail, or physically threaten users into explaining system details. Organizations often experience compromise due to internal threats and it is in the interest of world powers to seize enemy equipment, capture prisoners, and gather information through vast intelligence networks.

2.2 : The Key to Cryptography

During the early history of cryptography, as the secret mechanisms of cryptographic systems became known, and security by obscurity was invalidated, new systems would come to rely upon a variable cryptographic key. Cryptographic keys are of incredible importance to the cryptographic transformations that use them. Without the use of a variable cryptographic keys these algorithms can be trivially compromised. Cryptographic keys, typically manifest as unique bitstrings, are input parameters said to specify cryptographic transformation.(Van Tilborg & Jajodia 2011) All the operations involved in modern cryptographic message transformations are carried out in accordance with some algorithmic mechanism. This mechanism is parameterized by a secret key, which dictates the transformations to be enacted upon any plaintext message. Using a unique secret key for each execution of a cryptographic algorithm allows all plaintext values, even when initially identical, to undergo a unique set of transformations.(Gergersen 2017) Without a variable key, cryptographic methods proceed through an unchanging set of transformations to produce identical output ciphertext for identical input plaintext, known as deterministic execution.(Van Tilborg & Jajodia 2011) To invert such a transformation, adversaries only need knowledge of the algorithm used as it's execution would be deterministic, each transformation identical, and inversion straightforward.(Van Tilborg & Jajodia 2011) When a variable key is incorporated, a unique set of steps produces unique output based upon the key value. Thus, cryptographic keys are input parameters which determine the output of cryptographic algorithms. In the discussion that follows, we shall see that cryptographic keys, as miniscule bit sequences, are not only far more easily concealed than the entirety of a cryptographic algorithm, but are simple to change once compromised. Modern examples include the PIN of a bank account, the code to an electronic gate, the sequence of a combination lock, the password associated with a username.

In 1883, Auguste Kerckhoffs wrote two famous articles summarizing six contemporary principles for the design of cryptographic systems. What follows is an approximate English version: (Fabien Petitcolas 1997 par. 3)

- The system must be substantially, if not mathematically, undecipherable;
- The system must not require secrecy and can be stolen by the enemy without causing trouble;
- It must be easy to communicate and retain the key without the aid of written notes, it must also be easy to change or modify the key at the discretion of the correspondents;
- The system ought to be compatible with telegraph communication;
- The system must be portable, and its use must not require more than one person;
- Finally, given the circumstances in which such system is applied, it must be easy to use and must neither stress the mind or require the knowledge of a long series of rules.

While these principles are no longer of complete relevance given contemporary computational capability, the second statement is still a fundamental concept known to cryptographers as Kerckhoffs's principle.(Van Tilborg & Jajodia 2011) Kerckhoffs's principle can be generally understood to mean, a system should remain secure when every detail but the cryptographic key is public knowledge.

As all advancement in the field, this principle has been independently confirmed by other experts. Examples of such occurrences include Shannon's Maxim "the enemy knows the system",(Van Tilborg & Jajodia 2011 p. 675) i.e., system design should hold the assumption that adversaries have full knowledge of a system's operation. Bruce Schneier uses Kerckhoff's principle to support the belief that all security systems must be designed with sufficient fault tolerance such that disclosure of system details does not prevent useability. "Kerckhoffs's principle applies beyond codes and ciphers to security systems in general: every secret creates a potential failure point. Secrecy, in other words, is a prime cause of brittleness—and therefore something likely to make a system prone to catastrophic collapse. Conversely, openness provides ductility."(Mann 2002 par. 47)

Any cryptographic security system crucially depends on secrets or it's cryptographic operations would be straightforward to reverse.(Van Tilborg & Jajodia 2011) Modern cryptographic applications may be implemented by hardware or software. If the security of any application depends entirely on implementation secrecy, a compromise of that secrecy cannot be recovered. A compromised cryptographic system is useless and requires an entirely new implementation. The time scale of such a process is immense; design, implementation, verification, distribution, and maintenance of a system, unique from the previous implementation, is required to thwart attackers. Conversely, disclosure of cryptographic keys necessitates automatic key generation to replace. This is the essence of Kerckhoffs's principle, the secrets maintained by a security system must be those least difficult to replace when compromised.

2.3 : Cryptographic Primitives

Cryptologic research is supported by the fields of mathematics, computer science, electrical engineering, communication science, information science, and physics.(Ahmed Al-Vahed 2011)(Van Tilborg & Jajodia 2011) Cryptographic application supports the safe use of the most advanced and vital modern technologies including chip-based EMV smart payment cards, digital currencies, e-commerce, user authentication, secure data storage, secure network connections, and remains the leading approach to maintain classified military and government intelligence. (Ahmed Al-Vahed 2011)(Van Tilborg & Jajodia 2011) Cryptographic applications are primarily expressed as branches of engineering. (Ahmed Al-Vahed 2011) Most typically, engineering applications contend with the passive, neutral forces of nature.(Ahmed Al-Vahed 2011) The fields of cryptography and security engineering deal with the active, malevolent opposition of adversaries and attackers.(Ahmed Al-Vahed 2011)

In recent decades, the field has expanded beyond preservation of confidentiality. Modern cryptologic methods include techniques for data confidentiality, integrity preservation, authentication, identification, and access control, non-repudiation, interactive proofs, and secure computation in general.(Menezes Van Oorshot & Vanstone 1997) Each of these objectives are achieved individually by the proper cryptographic primitives.

A cryptographic primitive is a function that performs a fundamental cryptographic operation. Such functions are manifest as computer algorithms. Cryptographic primitives compose the basic building blocks of modern cryptographic systems. As fundamental components, cryptographic primitives are the foundation upon which security tools of greater complexity depend.

Example Cryptographic Primitives(Van Tilborg & Jajodia 2011)

Symmetric key	provides message confidentiality using a single key for encryption and decryption.
Public key	provides message confidentiality using a key pair, a public encryption key and a private decryption key.
One-way hash functions	provides message integrity by computing a unique hash value.

A general taxonomy of cryptographic primitives is given by Figure 3 below.

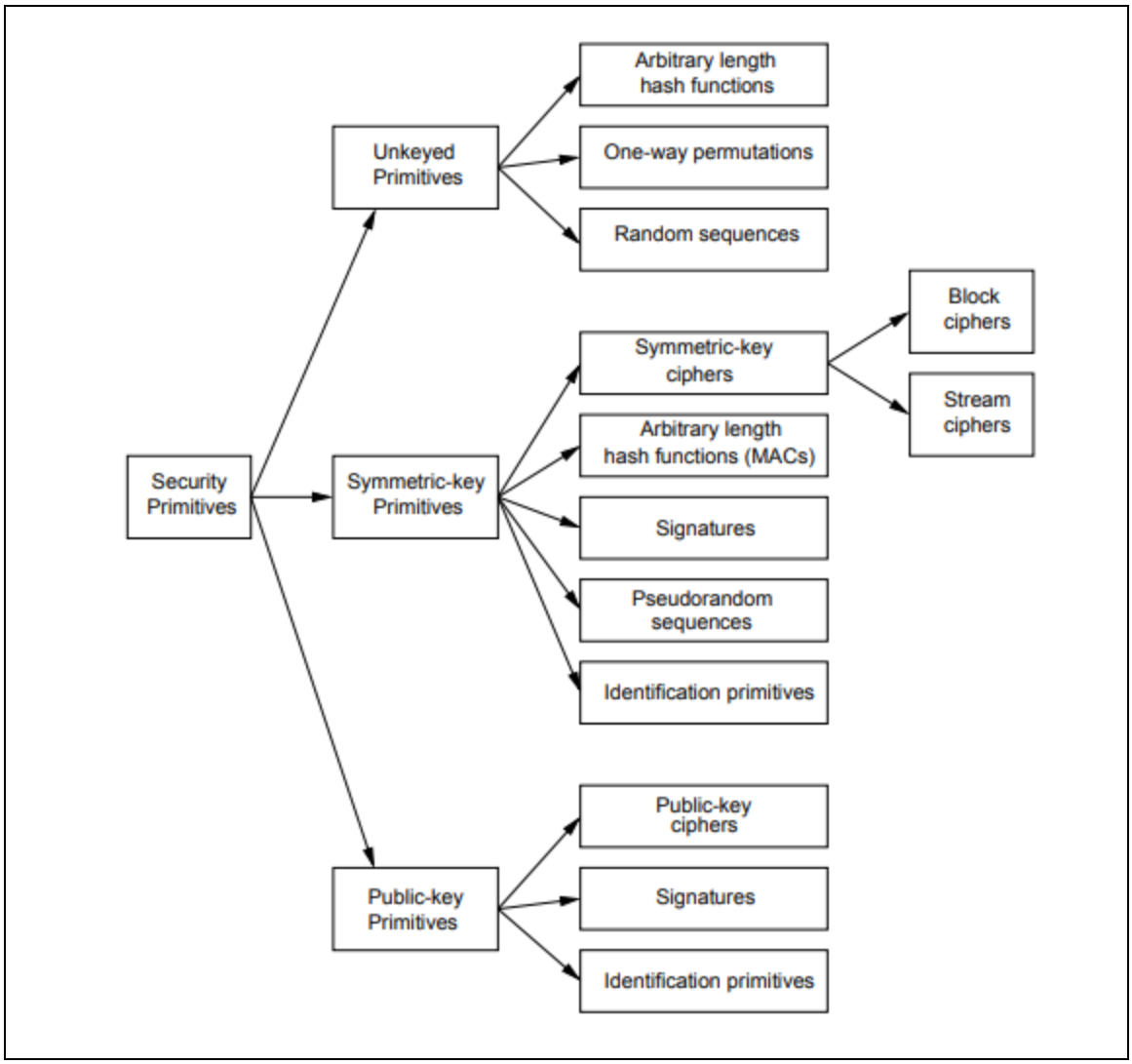

Figure 3

Each primitive must be profoundly reliable, performing in exact accordance to their specification. Due to this critical necessity, a significant amount of cryptologic research concerns the design and function of cryptographic primitives. Creation of cryptographic primitives is a complicated process; design, development, and verification to establish dependability takes considerable time. Among the reasons are Insufficient experience anticipating the theoretical and practical considerations involved, time consuming and error prone design processes, even for content experts, as well as the fact that completed algorithms require thorough, rigorous testing by the cryptological community.(Lafourcade 2013)

Successfully withstanding third party review gives some confidence that an algorithm is indeed secure enough for practical use.(Lafourcade 2013) Currently, extensive public review and cryptanalysis are currently the only method by which we can achieve a sufficient level of confidence, exhaustive proofs of security for an entire system are generally not feasible.(Lafourcade 2013) As such, it is highly insecure and resource inefficient to implement our own cryptographic primitives. "Several cryptographic primitives thought to be proven secure by their authors have been broken several years later, due to errors in the original proofs".(Lafourcade 2013 p. 7) This is of critical importance to security as any cryptographic system or protocol found to include a faulty cryptographic primitive would be, consequently, susceptible to attack.

Cryptographic system designers rely on publicly reviewed primitives, designed to precisely execute a single, specialized cryptographic operation. Cryptographic primitives are quite limited as each exactly defines one specific function. One or more cryptographic primitives are linked to develop processes of greater utility and complexity, known collectively as a cryptographic system, or *cryptosystem*.(Van Tilborg & Jajodia 2011) Cryptosystem structure often involves exchange of secure messages.(Van Tilborg & Jajodia 2011) Such cryptosystems are called cryptographic protocols as they define a method of exchange between two or more entities which fulfill a specified security objective.(Van Tilborg & Jajodia 2011) While cryptographic primitives and protocols both define methods of satisfying a security objective, primitives describe actions taken by one entity, whereas protocols describe the exchange between multiple.(Kotzanikolaou & Douligeris 2006) An example of a relevant cryptographic protocol is Transport Layer Security (TLS), a cryptographic protocol used to secure web (HTTPS) connections.(Van Tilborg & Jajodia 2011) Cryptographic protocols are used as cryptographic system components. In general, a cryptographic system consists of multiple cryptographic primitives and/or cryptographic protocols.(Van Tilborg & Jajodia 2011) In its entirety, a cryptosystem is a relationship consisting of an encryption method, a decryption method, and a well-defined sets of related plaintexts, ciphertexts, and cryptographic keys.(Van Tilborg & Jajodia 2011)

Cryptosystems provide advanced functionality to guarantee the complex security assurances necessitated by modern security requirements. (Van Tilborg & Jajodia 2011) Only when combined within a well-defined security system, can we achieve more than a single, simultaneous security requirement.(Van Tilborg & Jajodia 2011) Cryptographic systems, when constructed from well audited primitives, are able to assure many high-level security objectives in concurrency.(Van Tilborg & Jajodia 2011) These systems are only as secure as the cryptographic primitives which underlie them.(Van Tilborg & Jajodia 2011) Due to the vast complexity of modern cryptosystems, this paper focuses on the context, mathematical properties and implementation of a single cryptographic system.

2.4 : The One Time Pad

The first cryptographic function proven secure through the application of Information Theory was the One-Time Pad (OTP).[(Shannon 1949)] The OTP system input requires a message of arbitrary length and a key of equal or greater length, a transformation is then performed as each plaintext unit is combined with the key unit of corresponding location to produce the ciphertext unit.[(Shannon 1949)] First devised by Frank Miller for use with telegraph communications in 1882,[(Bellovin 2011)] this system was individually redeveloped in 1917 by Gilbert S. Vernam. On July 22, 1919, Vernam was issued U.S. Patent 1,310,719 for the "Secret Signaling System".[(Vernam 1919)] Now known as the Vernam cipher, this method specifies use of the XOR operation to encrypt a one-time pad.[(Vernam 1919)] The US National Security Agency (NSA) considers this patent "perhaps one of the most important in the history of cryptography".[(Klein 2003 p. 3)] However, in this original form, the system was insecure. Cipher operation combines plaintext units of a given location with key material units at the same location read from a punched tape. This key material tape was initially designed as a loop that was reused whenever it's end was reached. Joseph Mauborgne, a US Major General and the Army's 12th Chief Signal Officer, in command of the Signal Corps, would work with Vernam to introduce the final "one-time" system with a key that was both totally random and at least as long as the plaintext.[(Kahn 1996)]

How does the system function? We first suppose Alice wishes to send a message to Bob. To use the OTP, Alice must choose a key, typically from some collection of previously generated one time pads. Bob must know which key she selects, a process formally called key establishment, we will address this later in the document. Second, each unit from the pad must be combined with the plaintext unit of the same position to produce the ciphertext unit at that given position. The message units in this example will be letters of the english alphabet. The combination technique selected for this demonstration assigns each letter a numerical value, e.g., "A" is 1, "B" is 2, and so on until "Z" is 26. The numerical values of message and key letters of equal position are added together, such that the first plaintext unit with the first key unit through until the last plaintext and key unit are individually combined. As there are only 26 letters of the alphabet any result over 26 has 26 subtracted from it, the inverse operation, to produce a valid result. This way, if combination values go past the end of our alphabet, the sequence returns to A. This system is displayed by Table 7, below.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	...
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	1	...

Table 7

The resulting numerical value of the combination of a plaintext unit and key unit at a given position then corresponds to the output value the ciphertext digit at the same position. If the selected pad contains the key material "WPKLM", encryption of the message "HELLO" proceeds as shown by Figure 4, below.

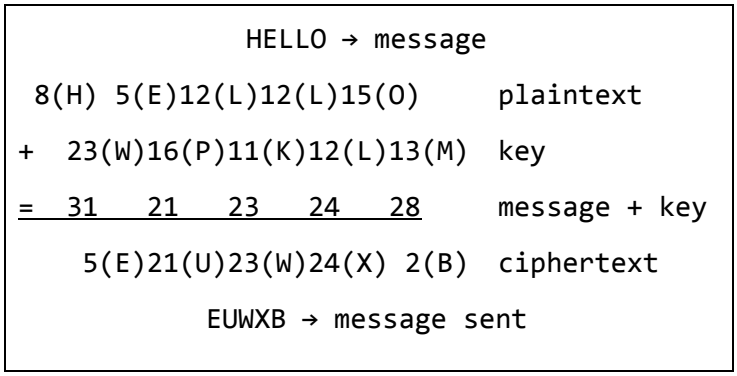

Figure 4

Alice is now able to send the secure message "EUWXB" to Bob. This ciphertext could be stored until later or be decrypted when received. Now suppossing Bob wishes to decrypt Alice's message, he must select the same one time pad as was input by Alice. Once this is known, Bob uses the matching key and the inverse combination process. To obtain the plaintext, the key is subtracted from the ciphertext and, if the resulting plaintext value is not positive, 26 is then added to produce the correct plaintext value. With proper application, Bob successfully recovers Alice's plaintext, the message reads "HELLO", shown by Figure 5, below.

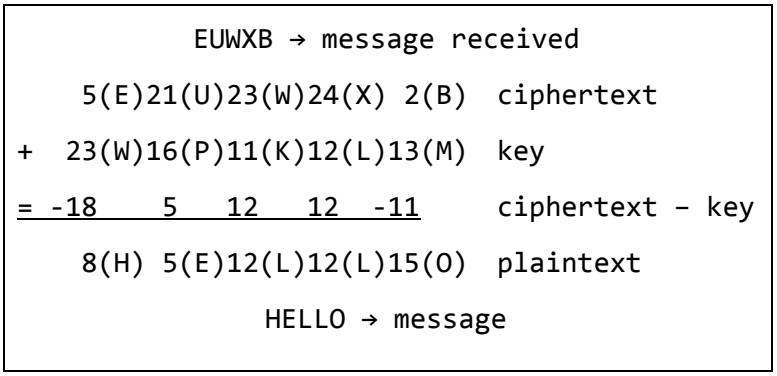

Figure 5

2.5 : Perfect Secrecy

The utility of this system becomes evident when we attempt to assess its security from an adversarial perspective. Suppose an eavesdropper, Eve obtains the previous message, "DTVWA". Given time significant time, Eve will surly discover key "WPKLM" produces plaintext "HELLO", however, the key "RSBRI" would also be found to produce plaintext "LATER", as shown by Figure 6, below.

```
                DTVWA → message
    4(D)20(T)22(V)23(W) 1(A)  ciphertext
–  18(R)19(S) 2(B)18(R) 9(I)  possible key
= –14    1   20    5   -8     ciphertext - key
=  12(L) 1(A)20(T) 5(E)18(R)  possible plaintext
            LATER → possible message
```

Figure 6

As each plaintext unit is combined with an individual, random key unit, the result of each combination operation is also independent. With sufficiently random key material, there is no information in the individual characters or whole of the ciphertext that will allow Eve to choose between all possible ciphertexts. Any message with the same number of characters is possible decryption output. Though many security practitioners intuitively understood this characteristic of the one time pad, it would not be until the 1940's that the mechanism by which the one-time pad achieves its security was mathematically proven.(Shannon 1949)

Claude E. Shannon, known as "the father of the information age"(GBCGMV 2002 p. 10) and for developing the study of information theory,(GBCGMV 2002) was the first to provide a mathematical definition for the function of cryptographic operations with "perfect security".(Shannon 1949 p. 659) While working for Bell Telephone Labs, Shannon published a classified memorandum in 1945, "A Mathematical Theory of Cryptography". The *Bell System Technical Journal* published a declassified version in 1949, "Communication Theory of Secrecy Systems".(Shannon 1949) Many experts credit this article as the advent of modern, mathematically validated cryptography.(GBCGMV 2002)

Shannon was inspired during World War II to address "[t]he problems of cryptography... ... secrecy systems furnish an interesting application of communication theory".(Shannon 1949 p. 1) Shannon's observations concerning communication theory and cryptography developed concurrently, that "they were so close together you couldn't separate them".(Kahn 1996 p 439) Shannon identified secrecy systems of two fundamental types.(Shannon 1949) One category included those systems designed to protect against adversaries with theoretically infinite resources, time, cryptanalysts, and, computational power, to reverse cryptographic operations.(Shannon 1949) This category Shannon called theoretical secrecy, now defined as unconditional security.(Shannon 1949)(Van Tilborg & Jajodia 2011) The other category is composed of systems designed to protect against adversaries with finite resources, what Shannon called practical secrecy, is now defined as computational security.(Shannon 1949)(Van Tilborg & Jajodia 2011)

Most of Shannon's work focused around theoretical secrecy as it was he who proposed the definition of a cipher with information theoretic or "perfect security".(Shannon 1949 p.659)After Shannon, If a cipher was information theoretic it was determined to be "unbreakable", an invaluable quality to be sure.(Reuvers & Simons 2013 par 1) Shannon determined that an encryption operation has perfect secrecy if the probability that encryption algorithm $E()$, when parameterized with message m_1 and key k, produces a given ciphertext c is equal to the probability that encryption algorithm $E()$, when parameterized with any other message m_2 and k, produces c.

$$P[E(m_1, k) = c] = P[E(m_2, k) = c]$$

Thus perfect secrecy means any two messages are equally as likely to correspond to given a ciphertext.(Shannon 1949) This is because, if *E* is a perfectly secure encryption function, for any fixed message *m*, there must be, for each ciphertext *c*, at least one key *k* such that $c = E(m, k)$. If this is true, an eavesdropper or other adversary can recover no information about the underlying plaintext once the transformation has been made, as all inverse transformations are equally likely options.(Shannon 1949)

2.6 : Key Issues With The One Time Pad

It has been claimed that information theorist Vladimir Kotelnikov had independently proven perfect security in the Sovite Union.(Holden 2017) His results, rumored to have been the subject of a 1941 report which remains classified.(Holden 2017) Shannon first delivered his results in 1945, publishing them in 1949.(Shannon 1949) Since Shannons time, one time pad ciphers have been used to secure critical communications, but issues inherent to their operation make these systems cumbersome.(Paar & Pelzi 2009) Use of the system requires assumptions that we will explore in more depth now.

As a result of Claude Shannon's revolutionary observations, it was proven that perfect secrecy is attainable using keys the same requirements as OTP keys.(Shannon 1949) This means we now know that perfect secrecy can only be obtained with a totally random secret key whose length, in text units, is greater than or equal to the amount of information being encrypted.(Shannon 1949) The exact transformation achieved by the OTP, if correctly parameterized. Such parameterization is also known as a "one time pad" or simply a "pad", i.e. any key fit for use in an OTP transformation.(Menezes Van Oorshot & Vanstone 1997) A pad has strict requirements each of which is a critical matter to the overall system security. If these requirements are not met, the cryptographic operation no longer meets the strict definition of the One Time Pad. Such an operation no longer possesses the security assurances associated with the use of the OTP.(Shannon 1949)

As we have discussed, if the key is at least the length of the plaintext, as well as perfectly random, used only once, and kept secret, then the system is "unbreakable".(Reuvers & Simons 2013 par 1) This key length and perfect randomness allows each unit of the plaintext to be encrypted, or ciphertext decrypted, by independent combination with the corresponding unit from the pad.(Menezes Van Oorshot & Vanstone 1997) This combination uses a randomness preserving operation, such as XOR patented by Gilbert Vernam, which we explore later in this document, in Section 5.5. Independent combination of the input units results in the equal probability for all possible inversion transformations.(Menezes Van Oorshot & Vanstone 1997) Thus, one time pads provide a perfectly secure transformation of arbitrarily sized messages so long as the key material is completely random, one time use, pre-shared, secret, of the same size as, or longer than, the message being transformed.(Shannon 1949)

The previous examples assumes two pads, identical sequences of random letters, were securely issued to both parties. This process is known as key distribution.(Van Tilborg & Jajodia 2011) The first modern symmetric cryptosystems used physical key distribution.(Van Tilborg & Jajodia 2011) Methods of the time typically involved concealable pads of paper and a pencil to perform the necessary

transformations. Cryptographic key material was centrally generated and stored on physical media such as paper or magnetic tape.(Van Tilborg & Jajodia 2011) Physical distribution to was accomplished through couriers, humorously referred to as “sneaker net”.(Van Tilborg & Jajodia 2011 p. 684) The KGB, Russia's intelligence organization, is famous it's one-time pads, so small they could fit in the palm of a hand, or in a walnut shell.(Smith 2007) As well, the security requirements of the OTP system dictates that all parties involved in secure message exchange destroy key material after use.(Reuvers & Simons 2013) This not only prevents reuse but also ensures key material does not fall into the wrong hands.(Reuvers & Simons 2013) The KGB is also famous for its solution to this problem, their one-time pad key material was printed on highly flammable nitrocellulose sheets which burn quickly, without ash.(Hannan & Asif 2017)

When implemented properly, one-time pads have the strongest possible security guarantee.(Houtven 2013) It would appear that the OTP solves the problem of perfect encryption and the cryptographic research of the last 70 years has been rather redundant.(Houtven 2013) This is not the case, use of one-time pads is rare due to being "horribly impractical".(Houtven 2013 p. 29) Application of the one time pad system requires the secure distribution of key material equal in length to the messages intended for encryption.(Reuvers & Simons 2013) This poses a problem for message exchange of nontrivial size or frequency. One time pads are not generally practical as it is difficult to distribute enough key bits to protect all future messages, use the proper key bits during encryption and decryption, avoid reuse of key bits by mistake, and keep all data involved secret.(Smith 2007) In retrospect, the OTP poses a trade-off: an information-theoretic security guarantee or impractical key requirements.(Houtven 2013) The One time pad marks the advent of modern cryptography, when the mathematical principles of information theory, discovered by Shannon, would mature and become coupled with new computational abilities. Theoretical and technological advancements since have made large strides in reducing the complexity of these problems. The mid-1970s saw two major public (i.e., non-secret) advances. The first advancement in modern cryptography provided a technological leap, manageable key sizes,(Kotzanikolaou & Douligeris 2006) as well as a surge in public awareness which revolutionized the process by which ciphers were designed. The second addressed the key distribution problem through key exchange protocols.

Section 3 : Symmetric Key Cryptography

3.1 : The Data Encryption Standard

The first modern cryptographic advancement since Shannon's contributions began with work at IBM in the early 1970s and culminated with the 1977 publication of the Data Encryption Standard (DES) U.S. Federal Information Processing Standard 46-3 (FIPS 46-3).(Branstad 1978) The ubiquity of digital communication and the rise of computer systems in the 1960s brought with it a need for both public and private sector security services, primarily methods of data protection.(Branstad 1978) This need was identified by a US standards bureau who brought it to public notice,(Branstad 1978) initiating, for the first time, a broad interest in cryptographic research.(Burr 1977)

To maintain security for its business associates and large financial organizations, the US National Bureau of Standards (NBS) conducted a study on the US government's digital communication capabilities in 1972.(Branstad 1978) Through this study NBS recognized a standard for encrypting sensitive information was necessary.(Branstad 1978) "Among the needs for physical, administrative and technical security measures and procedures, the need for a method of protecting computer data during transmission and storage was identified".(Branstad 1978 p. iv) In an effort to develop such capability, NBS worked with the National Security Agency (NSA) to determine criteria for a new national standard, soliciting proposals in 1973.(Branstad 1978) None of the submissions were suitable.(Branstad 1978)

August 1974 marked the beginning of the second round of solicitation, a research group at IBM was prepared for the call.(Branstad 1978) This triumphant contender was designed from 1973-74 at IBM, the algorithm was derived from Horst Feistel's work on the Lucifer cipher.(Keliher 1997) The draft Data Encryption Standard (DES) was published in the U.S. *Federal Register* on 17 March 1975.(Burr 1977) For the first time in the development of an encryption process, public comments were requested.(Branstad 1978) More amazingly this request was on a national scale, going out to those of industry, government experts, and the cryptographic community at large.(Burr 1977) Two public workshops were held to discuss the standard's selection via the design criteria(Burr 1977) before it was approved as a federal standard in November 1976.(Menezes Van Oorshot & Vanstone 1997) Final publication occurred on 15 January, 1977(Menezes Van Oorshot & Vanstone 1997) "The Standard specifies an algorithm for use by Federal Departments and Agencies in the cryptographic protection of unclassified computer data during transmission or in storage".(Branstad 1978 p. iv) This standard was made mandatory for all electronic fund transfer conducted by U.S. government including those of member banks of the Federal Reserve System.(Burr 1977)

Throughout the course of its application, cryptology has been a secretive practice. By example, it took decades before the declassification of the cryptanalytic principles of Japanese and German cipher machines from World War II.(Simmons 2009) However, the DES, from the moment of its inception, was a completely public algorithm.(Simmons 2009) "Every detail of its operations—enough to permit anyone who wished to program it on a microcomputer—was widely available in published form and on the Internet". (Simmons 2009 par 5) The result was that one of the best cryptographic systems in history was also the most public, a result that highlights the weakness of security by obscurity.(Simmons 2009)

Not only was DES the first publicly accessible cipher to be approved by a national agency, its subsequent adoption by standards organizations worldwide caused the DES to become the "de facto"(Simmons 2009 par. 3) international standard for both business and commercial data security. NBS' publication of the DES specification created a surge of public interest in cryptography which acted as the impetus for the first widespread interest academic cryptology, particularly cryptanalysis of block ciphers.(Burr 1977)

According to a NIST retrospective about DES:

> "The DES can be said to have "jump-started" the nonmilitary study and development of encryption algorithms. In the 1970s there were very few cryptographers, except for those in military or intelligence organizations, and little academic study of cryptography. There are now many active academic cryptologists, mathematics departments with strong programs in cryptography, and commercial information security companies and consultants. A generation of cryptanalysts has cut its teeth analyzing (that is, trying to "crack") the DES algorithm. In the words of cryptographer Bruce Schneier, "DES did more to galvanize the field of cryptanalysis than anything else. Now there was an algorithm to study." An astonishing share of the open literature in cryptography in the 1970s and 1980s dealt with the DES, and the DES is the standard against which every symmetric key algorithm since has been compared."(Burr 1977 p. 252)

3.2 : Symmetric Key Definitions

Symmetric-key cryptography was the only method of encryption publicly known before 1976.(Van Tilborg & Jajodia 2011) Before this time, all pre-modern, key dependent, cryptographic system implementation, whether simple transformations by hand or the intricate electromechanical machines used in World War II, were of the same logical class. Systems of this kind are known as secret-key, single-key, shared-key, one-key, private-key, or symmetric key cryptosystems.(Van Tilborg & Jajodia 2011) Symmetric systems require the sender and receiver use an identical cryptographic key during message transformation.(Van Tilborg & Jajodia 2011) Symmetric key algorithms are implemented as either a block cipher or a stream cipher.(Van Tilborg & Jajodia 2011)

Stream ciphers transform individual plaintext digits sequentially.(Van Tilborg & Jajodia 2011) The size of plaintext digits are determined by the stream cipher method.(Paar & Pelzi 2009) Dated methods like the Caesar or Vigenere ciphers, process single letters, while modern implementations, like RC4, process single bits.(Paar & Pelzi 2009) In general, stream cipher output is dependent on a hidden internal state, initialized by cryptographic key, which changes as the cipher operates.(Stallings 2017) A stream cipher algorithm functions by first using the key material to generate the "keystream", a stream of pseudorandom digits of arbitrary length the same as, or longer than, that of the plaintext to be encrypted.(Van Tilborg & Jajodia 2011) Keystream and plaintext digits are then combined, via a randomness preserving operation, to produce a seemingly random digit of the ciphertext stream.(Van Tilborg & Jajodia 2011) The inverse operation uses the same cryptographic key to generate an identical key stream that is combined with the ciphertext to recreate the original plaintext.(Van Tilborg & Jajodia 2011)

As with a One-Time Pad, key stream units are sequentially combined with plaintext units to produce a unit of the ciphertext. In this way, a stream cipher emulates the function of the OTP which is proven to have perfect secrecy.(Van Tilborg & Jajodia 2011) However, the key stream is pseudorandom, algorithmically generated from a small, fixed size cryptographic key.(Van Tilborg & Jajodia 2011) Modern examples range from 64 - 256 bits.(MHFP 2015) We know, from the earlier OTP discussion, that a cryptographic cipher designed to achieve perfect secrecy must have a key as long as the intended plaintext message.(Shannon 1949) Using the cryptographic key as input, a stream cipher algorithmically generates a pseudorandom keystream of arbitrary length as long or longer than the plaintext.(Van Tilborg & Jajodia, 2011) It is this keystream that is combined, using a randomness preserving operation, with the plaintext digits.(Van Tilborg & Jajodia, 2011) As the key of a stream cipher is less than the length of the plaintext it can no longer guarantee perfect secrecy, the resulting keystream is only pseudorandom.(Shannon 1949) While stream ciphers can achieve high security assurance, the proof of security associated with a one-time pad no longer hold.

Converse to the individual transformations applied by stream ciphers, block ciphers transform fixed-length groups of bits, the same size as the key, called blocks.(Van Tilborg & Jajodia, 2011) Block cipher output is generated by a deterministic transformation that is specified by a symmetric key the same size as the block intended for encryption, similar to an OTP.(Van Tilborg & Jajodia, 2011) However, the size of the block is determined by the cipher and while a cipher may be capable of operation on more than one block size, prior to encryption, plaintext messages must be padded.(Van Tilborg & Jajodia, 2011) Padding adds artificial plaintext material until messages are a multiple of the block size selected during a particular operation. Methods of padding are explained later, in Section 10.2, of this report. Next, we explore the fundamental concepts which connect Shannon's discoveries in Information Science to the revolutionary DES and its contemporary successor, the Advanced Encryption Standard (AES).(Stallings 2017)

3.3 : Symmetric Block Cipher Design

This section will explore the concepts of diffusion, confusion, and how the operations that provide them relate to the evolution of cipher structure. Confusion and diffusion were first identified as properties of a secure cipher identified by Claude Shannon in *A Mathematical Theory of Cryptography(1945)*.(Shannon 1945) To paraphrase Shannon, confusion causes the relationship between the ciphertext and the symmetric key to be as complex as possible, where as diffusion dissipates the statistical structure of the plaintext over the entirety of ciphertext.(Shannon 1949) In his work, Shannon suggests that a combination of the two transformations are sufficient to obscure the structural characteristics of the plaintext and impede statistical analysis of its relation to the key and ciphertext.(Shannon 1949) Today, the properties of confusion and diffusion have quantifiable definitions.

Confusion applies to ciphers where by each ciphertext unit has highly nonlinear relations with multiple key bits.(Stallings 2017) A function said to provide confusion generally means a process changes data from the input to the output in a complex way which obscures their relationship.(Keliher 1997) In the context of cryptographic ciphers, confusion attempts to make discovery of the key used for a particular cryptographic transformation as difficult as possible.(Stallings 2017) Even by cryptanalysis of a large number of related plaintext and ciphertext, the relation between the key material input and the ciphertext output must be so complex as to make it relatively impossible to deduce.(Stallings 2017) Therefore, by a function with high confusion, each ciphertext bit should depend on the entire key, as well as in distinct ways on various key bits such that variance in even a single key bit alters the transformation entirely.(Stallings 2017) Modern methodology recommends this be "...achieved by the use of a complex substitution algorithm".(Stallings 2017 p. 125)

Diffusion applies to ciphers where, statistically, a single plaintext bit flip causes a change in half of the ciphertext bits.(Stallings 2017) A function said to provide diffusion generally means that changes in a plaintext unit will reflect in a quantifiable portion of the output.(Stallings 2017) In the context of cryptographic ciphers, diffusion attempts to make the statistical analysis relationship between the plaintext and ciphertext as complex as possible.(Stallings 2017) "This is achieved by having each plaintext digit affect the value of many ciphertext digits; generally, this is equivalent to having each ciphertext digit be affected by many plaintext digits."(Stallings 2017 p. 124) Done well, every plaintext unit affects every unit of the ciphertext, complicating cryptanalysis.(Stallings 2017) Even when an adversary has intercepted sufficient material, the cryptanalytic work required is great, as the statistical relationships of the plaintext are diffused, evident only in blocks of very small individual probability.(Shannon 1949) "In a binary block cipher, diffusion can be achieved by repeatedly performing some permutation on the data followed by applying a function to that

permutation".(Stallings 2017 p. 125) When this is done, plaintext bits of different positions contribute to each ciphertext bit.(Stallings 2017)

Generally, the simplest way to achieve diffusion and confusion properties is through a well-defined and repeatable series of substitutions and transpositions.(Shannon 1949) All ciphers involve transposition or substitution in some form, and those that employ a combination of these two mathematical operations are capable of particularly robust security assurances.(Britannica 2016) Substitution replace one symbol, or symbol group, with another symbol, or symbol group. A cryptographic substitution is an operation that replaces a plaintext unit or segment value with a defined ciphertext value.(Van Tilborg & Jajodia, 2011) Transposition exchanges the location of two entities.(Van Tilborg & Jajodia, 2011) The strict cryptographic definition requires that only two members of a given set are transposed while all other members retain their location. In practice the term transposition is often used erroneously to reference an arbitrary reordering of set members, a function defined as permutation. The reason for their interchangeability stems from the fact that a series of transpositions can be made to equal any permutation. As is convention, we will use permutation to generally reference such transformations when we do not require the strictness of transposition. We can compare cryptographic substitution and transposition functions. Cryptographic transposition exchanges a plaintext unit's location without altering the value.(Britannica 2016) Cryptographic substitution alters a plaintext unit's value without a change in location.(Britannica 2016)

For transformations involving a nontrivial number of message symbols both functions are individually insecure.(Shannon 1949) Modern cryptographic round functions thus implement substitution operations with sufficient properties of confusion and transposition operations with sufficient properties of diffusion.(Menezes Van Oorshot & Vanstone 1997) Modern block ciphers consist of an initial conversion to binary before applying cipher rounds with a well defined sequence of S-box substitutions and P-box permutations.(Menezes Van Oorshot & Vanstone 1997)

A substitution box or S-box is used to substitute a block of input bits for a corresponding block of output bits.(Asif Buchanan Li 2018) S-Boxes use nonlinear Boolean functions to obscure the relationship between the key and the ciphertext, achieving confusion.(Van Tilborg & Jajodia, 2011) The desired property is that each output bit will depend on every input bit.(Asif Buchanan Li 2018) A well designed S-box has the effect that a change one bit of plaintext will, on average, result in a change in half of the ciphertext bits.(Asif Buchanan Li 2018)

A permutation box or P-box is used to permute input bits.(Brown & Seberry 1990) A P-Box causes diffusion by individual transposition of the outputs of one round forming a permutation that is input to the next round.(Brown & Seberry 1990) An effective P-box ensures the bits of any input source are distributed to as many individual inputs of the next rounds.(Brown & Seberry 1990) In modern ciphers

this is done such that each of the S-box input bits come from the outputs of different S-boxes and none of the input bits to a given S-box comes from the output of that same S-box.(Brown & Seberry 1990)

The substitution of binary units is a form of fractionation.(Britannica 2016) Fractionation is a general substitution of individual symbols in the plaintext to multiple symbols in the ciphertext,(Britannica 2016) the method of encoding implemented by Francis Bacon in his Bacon Cipher. Modern fractionation methods convert each message unit by a standardized binary representation. The binary string created undergoes cryptographic transformation, the result of which may then be reverted into a message alphabet for human interpretation. When a fractionated message is transposed, the components of message units become widely separated in the cipher text, achieving diffusion.(Kopal 2018)

A cipher alternating application of substitution and permutation transformations was first implemented by Horst Feistel, who lead a team at IBM in the late 1960's.(Keliher 1997) Feistel's famous Lucifer cipher is a descendant of Shannon's product cipher that alternates confusion and diffusion functions.(Stallings 2017) It was found that iterating a combination of these functions on the binary string generally makes cryptanalysis increasingly harder to leverage.(GJMN 2015) This structure became the archetype for block cipher design after DES was adopted as the US national cryptographic standard.(Keliher 1997) To this day, block cipher design relies on confusion and diffusion of message structure. The Feistel cipher structure, nearly half a century in age, is the general structure implemented by Triple Data Encryption Algorithm (TDEA) and the Advanced Encryption Algorithm (AEA) the two encryption algorithms currently approved for protecting unclassified computer data by NIST.(Stallings 2017)

Most contemporary block ciphers are categorized as iterated product ciphers.(Van Tilborg & Jajodia, 2011) Modeled after the concept of a product cipher, systems built by composition of simple cryptographic operations.(Van Tilborg & Jajodia, 2011) The Product Cipher was first put forth by Claude Shannon in *Communication Theory of Secrecy Systems(1949)*.(Shannon 1949) When properly designed, the resulting product cipher shows greater resilience under cryptanalysis than the component operations.(Shannon 1949) Iterated ciphers use repeat applications of a round function, an invertible cryptographic transformation, to convert fixed-size blocks of message text.(Van Tilborg & Jajodia, 2011) Each cipher iteration consists of one application of the round function and each round function is composed of a sequence of cryptographic operations.(Van Tilborg & Jajodia, 2011) As cryptographic primitives generally compose a given cryptographic system, cryptographic operations are the basic building blocks of cryptographic cipher rounds.(Van Tilborg & Jajodia, 2011) A product cipher is called an iterative product cipher if all round functions are identical.(Van Tilborg & Jajodia, 2011)

Modern cipher round structure is defined by the concept of an SP-network, or substitution–permutation network (SPN).(Van Tilborg & Jajodia, 2011) An SPN takes a plaintext block and cryptographic key as inputs, applying cipher rounds, alternating substitution boxes (S-boxes) and permutation boxes (P-boxes), to produce the output ciphertext block.(Van Tilborg & Jajodia, 2011) Each round transformation is executed in the same manner, with variance provided by round values, generally called round constants a round key or subkey.(Van Tilborg & Jajodia, 2011) A key schedule algorithm calculates round keys through the use of simple cryptographic operations, such as S-boxes and P-boxes, on the input cryptographic key.(Van Tilborg & Jajodia, 2011) During each round, the round key is combined using what is known as a group operation, a binary operation satisfying certain mathematical axioms.(Van Tilborg & Jajodia, 2011) We will cover these later, in Section 6.2.

Two key properties of SP networks are the avalanche property, identified by Feistel; and the completeness property, identified by Kam and Davida.(Brown & Seberry 1990) Completeness effect applies whenever each bit of the ciphertext depends upon every plaintext bit.(Keliher 1997) Avalanche effect applies whenever one input bit is changed, on average half the output bits change.(Keliher 1997) These effects work to ensure that every output bit becomes related to of each input bit in as few rounds as possible.(Brown & Seberry 1990) A well-designed SPN implements alternating rounds of S-box substitutions and P-box permutations to satisfy the properties of confusion and diffusion to thwart application of statistical cryptanalysis.(Keliher 1997) An effective cryptographic transformation must redistribute non-uniformity of plaintext bits across much larger structures in the ciphertext, making that non-uniformity indetectable. As a bit can have only two states, a secure cryptographic transformation should function such that bit conversion, from one seemingly random state to another, occurs with half probability. In the output of a well designed SPN, statistically half of the bits are related to any one input bit.(Van Tilborg & Jajodia, 2011) Therefore the value of any single input bit is hard to predict.(Van Tilborg & Jajodia, 2011) However, one more fact that we have previously discussed, without a secure cryptographic key, the SPN, as well as all modern cryptographic methods, perform a complex but fully deterministic transformation of its inputs.(Van Tilborg & Jajodia, 2011)

3.4 : Key Management

Generally, cryptographic keys are the mechanism by which cryptographic algorithms allow information to remain secure when transmitted over untrusted channels or held in untrusted storage. In practice, keys are used as cryptographic system input to produce secure output. We have discussed how cryptographic keys are implemented such that they provide security even if the cryptographic method is known completely by adversaries. If a system is secure even when the enemy knows everything except the key, then all that is needed is to manage keeping the keys secret. By successfully managing keys, cryptographic methods replace a difficult problem, keeping an indefinite amount of information secure, all messages, with a much more manageable one, keeping a single relatively small secret secure, an encryption key. A system that requires long-term secrecy for something as large and complex as the whole of it's design obviously cannot achieve that goal. As we have discussed, those systems designed with obscurity as the primary mechanism of security assurance only replaces one difficult problem, the secrecy of all messages, with another of near equal difficulty, the secrecy of a widely applied process. Therefore, successful key management is critical to the security of any cryptosystem.

Key Management is generally the process of secure cryptographic key use in relation to a given cryptosystem.(Van Tilborg & Jajodia, 2011) Of primary concern for users wishing to exchange cryptographic messages, is the parameterization necessary to transform messages sent and invert messages received. To ensure security, users must establish and maintain those inputs necessitated by their intended cryptosystem. The life cycle associated with cryptographic keying material includes their generation, distribution, storage, update, and cancellation.(Van Tilborg & Jajodia, 2011) Key generation for example, ensures that cryptographic keys are both sufficiently random and contain enough entropy to prevent it from being guessed by adversaries or discovered through cryptanalysis.(Van Tilborg & Jajodia, 2011) This problem, while difficult, has been addressed in many ways by various cryptographic systems RFC 4086.(Schiller Crocker 2005) Key generation is primarily a mathematical and thus computational process. In contrast, other aspects of key management involve social, political, legal, and ethical considerations such as organizational practices and policies, user education, and coordination between individual, departmental, and external entities.(Van Tilborg & Jajodia, 2011) Due to the necessary inclusion of significant human involvement, key management is a deeply challenging aspect of cryptographic practice.(Van Tilborg & Jajodia, 2011) This document will not address every aspect of Key Management, rather, it hopes to highlight major complications and their solutions.

Historically, one of the most critical challenges in the practice of cryptography has been the secure distribution of keys.(Van Tilborg & Jajodia, 2011) Cryptographic keys, in practice, represent a shared secret between entities used to maintain a secure communication channel. The process of key distribution defines methods for exchange of the information necessary to establish a secure

communication channel.(Van Tilborg & Jajodia, 2011) The key exchange problem asks how entities of separate location can agree upon a cryptographic key, without risk of eavesdroppers.(Van Tilborg & Jajodia, 2011) If two parties cannot establish secure key distribution they won't be able to communicate without the risk of messages being interpreted by adversaries. Key exchange protocols generally define a method by which cryptographic key information is communicated between parties.(Van Tilborg & Jajodia, 2011) Secure key exchange is difficult as it must occur via a secure channel.(Van Tilborg & Jajodia, 2011) As key exchange is a form of information exchange it is said to transpire across a channel between communicating participants. A 'secure channel' generally references some method of transferring data resistant to eavesdroppers.(Van Tilborg & Jajodia, 2011) The act of key exchange is said to occur either "in-band" or "out-of-band".(Dulaney & Easttom 2017 p 242) In-band keys are exchanged through the same communication channel intended for encrypted information.(Dulaney & Easttom 2017) Out-of-band keys are exchanged via any other communication channel than the one intended for encrypted information.(Dulaney & Easttom 2017)

Perfectly secure channels do not exist in the physical world.(PSP, 2016) "There are, at best, only ways to make insecure channels (e.g., couriers, homing pigeons, diplomatic bags, etc.) less insecure: padlocks (between courier wrists and a briefcase), loyalty tests, security investigations, and guns for courier personnel, diplomatic immunity for diplomatic bags, and so forth. ".(PSP, 2016 p. 2) One time pads, though they produce perfectly secure ciphertext with a simple transformation, require the same amount of key information to be shared as the information to be encrypted. Modern ciphers retain much of the security offered by the one time pad system while requiring a smaller information exchange in the form of cryptographic keys.(Van Tilborg & Jajodia, 2011) As we have seen, instances where a symmetric key cryptosystem is used, require the exchange of an identical key as symmetric-key algorithms use the same key for encryption and decryption of a message. A severe disadvantage of symmetric, or single-key, cryptography is that it requires a secret key to be established between users and maintained in secret for secure use. This requirement is necessary as the proper cryptographic keying facilitates the successful operation of a given cryptosystem.

While symmetric keys are miniscule in length relative to those of the OTP, if capability of message exchange between users is meant to be both complete and secure, the sharing of a distinct key is required for each possible pair of communicating parties.(Kotzanikolaou & Douligeris 2006) We will find that, with each added participant, the required number of keys increases rapidly.(Kotzanikolaou & Douligeris 2006) We have discussed how one pair of entities would require a single key, what is required for each new participant? Each new user added must generate a key for each previous user. If we were to consider adding the nth unique participant, it would require $(n-1)$ new keys. Beginning from a pair, each key addition forms a series $1+2+...+(n-1) = n(n-1)/2$ keys are required.(Kotzanikolaou & Douligeris 2006) This leads to a need for 45 unique keys for 10 participants, 4950 keys for 100 and to 499500 keys for 1000.(Kotzanikolaou & Douligeris 2006) Not only must a secret key be securely exchanged between each pair of communicating entities prior to system use, as each key is shared between two entities, its

future secrecy and thus a system's security depends on both entities.(Kotzanikolaou & Douligeris 2006) Additionally, due to the difficulty of key distribution, duration of cryptographic key use, and thus the frequency of key updates further amplify the key exchange problem. As it increases adversarial effort to recover multiple keys, they should be frequently changed. Additionally, while groups of messages could share a key, ideal security requires keys be replaced after each ciphertext exchange. This practice limits risks involved with system failure, as key update frequency increases, number of ciphertexts recoverable on key compromise, decreases. However, the difficulty of consistent secure key use increases in proportion to the number of participants and their messages.

To understand modern key management, it is important to understand the two types of cryptosystems, symmetric or secret key and asymmetric or public key.(Van Tilborg & Jajodia, 2011) The theory of asymmetric methods was first made public in 1976.(Van Tilborg & Jajodia, 2011) The technology to support public use of these methods was not available before the mid 1990's.(Van Tilborg & Jajodia, 2011) All earlier cryptographic systems, both ancient and modern, were symmetric in nature. As we have discussed, symmetric cryptography is based on the use of a single secret key is used to perform both a cryptographic operation and its inverse.(Van Tilborg & Jajodia, 2011) This necessitates a secure method of conveyance for at least one cryptographic key copy and a heightened risk of compromise during transfer and at either end-point due to this shared key.(Van Tilborg & Jajodia, 2011) The requirement of shared secret key access is the defining disadvantage of symmetric systems.

Asymmetric or Public key cryptography is based on the use of a mathematically related key pair where by one member of the pair is used to perform a cryptographic operation and the other is used to perform the cryptographic inverse.(Van Tilborg & Jajodia, 2011) This pair consists of a key which may be readily published or revealed without risk, the public key.(Van Tilborg & Jajodia, 2011) As well as a key which must be kept secret or revealed only to trusted parties, the private key.(Van Tilborg & Jajodia, 2011) In practical systems, the mathematical relationship shared by an asymmetric key pair is made such that knowledge of the public key does not allow determination of the private key efficiently.(Van Tilborg & Jajodia, 2011) While public-key systems are capable of the functionality necessitated by modern security systems, it is not best practice to use them on their own.(Paar & Pelzi 2009) Some of the most prevalent misunderstandings related to asymmetric cryptography follow.

It is commonly thought that asymmetric encryption methods are less susceptible to cryptanalysis.(Stallings 2017) **"**There is nothing in principle about either symmetric or public-key encryption that makes one superior to another from the point of view of resisting cryptanalysis**"**.(Stallings 2017 p. 284) The security provided by a given encryption scheme is dependent firstly on the relative strength of it's design and then on cryptographic key size as it determines computational work involved in compromise by brute force.(Stallings 2017) The second is that asymmetric systems provide a comprehensive solution to the faults of, and thus make obsolete,

symmetric system use.(Stallings 2017) In fact, "because of the computational overhead of current public-key encryption schemes, there seems no foreseeable likelihood that symmetric encryption will be abandoned".(Stallings 2017 p. 284) In practice, public-key encryption methods are incredibly computationally intensive, about one hundred to one thousand times slower than private-key algorithms.(Paar & Pelzi 2009) Lastly, there is a feeling that the asymmetric solution of public key distribution makes trivial the cumbersome handshaking and exponential growth involved with symmetric secret key distribution.(Stallings 2017) In fact, " … some form of protocol is needed, generally involving a central agent, and the procedures involved are not simpler nor any more efficient than those required for symmetric encryption".(Stallings 2017 p. 285)

Symmetric cryptography and asymmetric cryptography are not mutually exclusive, these techniques are used to complement each other in practice.(Van Tilborg & Jajodia, 2011) Most practical protocols use a hybrid approach which incorporates both symmetric and asymmetric primitives.(Paar & Pelzi 2009) "Examples include the SSL/TLS protocol that is commonly used for secure Web connections, or IPsec, the security part of the Internet communication protocol".(Paar & Pelzi 2009 p. 154) Modern information security systems, among other things, use symmetric cryptographic primitives for the encryption and decryption of data and asymmetric cryptographic primitives for key distribution.(Van Tilborg & Jajodia, 2011) As a basic example, "symmetric cryptography can be used to encrypt a message and asymmetric cryptography can be used to securely transfer the secret key used to encrypt the file to the intended recipient(s)".(Van Tilborg & Jajodia, 2011 p. 684) Ironically, while it eliminates many of the problems associated with symmetric key distribution, asymmetric cryptography is rarely used for data encryption.(Paar & Pelzi 2009) As public-key inventor, Whitfield Diffie, has said, “the restriction of public-key cryptography to key management and signature applications is almost universally accepted.”(Stallings 2017 p. 284)

3.5 : The Advanced Encryption Standard

The 1990s brought the World Wide Web to the public, the excitement which ensued generated rapid economic expansion in computer technology.(LFS 2018) Serious threats arose in the 1990s, challenging the security of both governmental and commercial interests, and called for a reassessment of the US's cryptographic capability.(LFS 2018) Before then, sensitive information sent over the Internet, such as financial data, was encrypted, if at all, with DES.(LFS 2018) Due to its vast popularity and duration of use, the DES is the cryptographic system and de facto standard against which every symmetric key algorithm of the previous century was compared.(Burr 1977) Regardless of DES' popularity, the 56-bit key size was thought to be too small, even upon release in 1976.(Gilmore 2005) It was claimed that governments in the 1970's had sufficient computing power to break DES.(Gilmore 2005) By the 1990s close, cryptanalysts would recover DES messages in under 24 hours.(LFS 2018)

NBS, renamed the National Institute of Standards and Technology (NIST) in 1988, held another competition to determine a suitable standard, this time accepting international submissions.(LFS 2018) In the September 1997 Federal Register, NIST solicited submissions for the Advanced Encryption Algorithm, which would be “an unclassified, publicly disclosed encryption algorithm available royalty-free worldwide that is capable of protecting sensitive Government information well into the next century.”(LFS 2018 p. 17) NIST received 21 submissions and held the first public conference in 1998.(LFS 2018) The selection process was even more open and transparent than its predecessor. DES was officially replaced by the Advanced Encryption Standard (AES) after selection of Rijndael, the submission of Belgian cryptographers Vincent Rijmen and Joan Daemen.(LFS 2018) The selection process as well as the resulting cipher won praise from the international cryptographic community, presumably restoring confidence to those suspicious of a backdoor in the standard's predecessor.

In 2001 NIST announced FIPS 197 and DES' aging designation as a standard was withdrawn. Despite deprecation as an official standard, DES, specifically the still-approved and computationally secure triple-DES variant, remained popular.(LFS 2018) DES' 56-bit key-size was shown to be insufficient when, in 1997, RSA Data Security began to issue public challenges to break DES.(Almunawar 2001) The first was completed by a team in 96 days, two more challenges were hosted the next year, were broken in 41 days and then 56 hours. Six months later in January of 1999 DES was broken in 22.25 hours.(Almunawar 2001) As a result, the use of single DES encryption is now insecure and should not be included in new cryptosystem design, as well, messages protected by any cryptosystem implementing single DES are at risk.

Since standardization by the US federal government in 2001, the U.S. Department of Commerce’s National Institute of Standards and Technology (NIST) estimates a $250 billion

economic impact from the Advanced Encryption Standard (AES) adoption by private industry.(LFS 2018 "Today, the AES protects everything from classified data and bank transactions to online shopping and social media apps."(NIST 2018 par. 2) Professor Christof Paar of the Ruhr University of Bochum, Germany, an internationally-renowned cryptography and AES specialist, estimates that the algorithm encrypts well over half of all newly created data.(Chernev 2019) This paper seeks to provide information on the theory, context, and mechanism of the world's most used cryptographic system, Rijndael, the Advanced Encryption Standard.(Paar & Pelzi 2009)

Part 2

Mathematical Techniques of the Advanced Encryption Standard

Section 4 : Set Theory

Section 4.1 : Establishing Sets

To facilitate the understanding of cryptographic algorithms it is useful to examine relevant mathematics. Modern cryptographic primitives implemented in both symmetric and asymmetric ciphers are based on arithmetic within a finite number of elements.(Paar & Pelzi 2009) Number sets with which we are familiar, such as the set of integers, are infinite.(Paar & Pelzi 2009) Generally, sets are abstract structures, representing unordered collections of distinct objects.(Cohn 1981) Formally, the entities which comprise a given set define the collection and thus determine its uniqueness. (Cohn 1981) As each element of a set is distinct and the collection is unordered, how we choose to list or represent a set makes no difference, every representation is equivalent to the same abstract structure.

By convention a set A, is represented by capital letters and the elements which a given set contains are denoted by lower-case letters, a. If we suppose the set S, contains a given element e, we state that e is an element (or member) of S, or e is in S, or e belongs to S. We denote the relation "element of a set", known as set membership, by the "∈" symbol. We express the membership relation, $e \in S$. Conversely, if element e is not a member of the set S, we state that e is not an element of S, or that e is not in S. We express this relationship, $e \notin S$. Sets are potentially infinite collections, consisting of distinct objects said to possess the relation of membership. This membership relation is said to define the set.

A set is said to be finite if it possesses a limited number of member elements, and infinite otherwise.(Cohn 1981) The cardinality, or order, of a finite set S is the number of members in S, denoted |S| = n where n ∈ ℕ.(Cohn 1981) We can thus conceptualize the utility granted by a set's definition to derive from the fact that it divides an aggregate into two distinct collections: Those objects which are members of a set and those that are not. Due to the generality of the set structure, member elements can represent any group of abstract objects (i.e. numbers, colors, symbols, etc…) We have defined the convention and notation by which we specify a set S, an individual element x, and whether they possess the relation of membership, x ∈ S, or not x ∉ S. We now specify a set in its entirety, explicitly, implicitly, and by predicate.

We may explicitly define a finite set through the individual specification of its members.(PM 2015) Set members are bounded by braces { }, in an unordered, comma delimited sequence, $\{e, o, i, a, u\}$.(PM 2015) A set S may express the relation of equivalence to an explicitly defined set via the equality operator =, S = $\{e, o, i, a, u\}$.(PM 2015) Sets of considerable or possibly infinite

cardinality become impossible to represent in this way and we must rely on other conventions to define membership.(PM 2015)

If the elements in a set have an obvious pattern, we can define the set implicitly using ellipsis (…).(PM 2015) Suppose we have an explicitly defined set S where S = {1,2,3,4,5,6,7,8,9,10}.(PM 2015) An implicit definition might be S = {1,2,…,10}.(PM 2015) We are meant to observe the elements count up uniformly, and might read this definition as: S is the set containing 1, increasing by 1, to 10.(PM 2015)

Often we wish to define more abstract properties via membership relations.(PM 2015) An object specified via predicate, is defined in terms of a property that it possesses.(PM 2015) Whether an object x possesses a particular property P is either true or false, and so can be the subject of a propositional function P(x).(PM 2015) A set may then be specified by such a propositional function, e.g.: S={x | P(x)}.(PM 2015) This definition is to be interpreted as S is the set of objects which satisfy the property P.(PM 2015) In this context, we see that the vertical bar symbol "|" is interpreted to mean "such that".(PM 2015 par. 5) The previous definition could then be interpreted formally as, S is the set of all x, such that P(x) is true.(PM 2015) To provide further example, the definition S = { (x, y) | x, y ∈ ℕ } could be interpreted, S is the set of all ordered pairs (x, y) such that x and y are in the natural numbers.(PM 2015) Be aware, some texts prefer a colon ":" instead of a vertical bar.(PM 2015)

There are also sets so common that they are given their own symbology, examples used throughout this document include:(Jackson 2017)

The integers, denoted $\mathbb{Z}$,	{... , −2, −1, 0, 1, 2,...}
The positive integers, denoted $\mathbb{Z}^+$,	{1, 2, 3,...}
The non-negative integers less than n, denoted $\mathbb{Z}_n$,	{0, 1, …, (n-1)}.

Section 4.2 : Operations on Sets

Sets are often used to model mathematical operations. Generally, an operation is a function f, of the form $f : X \to Y$, where X is the domain set, Y is the codomain set, and f models a transformation or mapping between the two sets.(Jackson 2017) An operation's domain is the set for which the function is defined to produce output.(Jackson 2017) An operation's codomain is the set within which output is constrained to fall.(Jackson 2017)

An Injective or one to one function, maps a domain member to no more than one codomain member.(Jackson 2017) An injection is denoted:

$$f : X \to Y \ \forall\, x, x' \in X,\ f(x) = f(x') \Rightarrow x = x' \ .$$(Jackson 2017)

A surjective or onto function maps domain members such that at least one corresponds to each codomain member.(Jackson 2017) A surjection is denoted:

$$f : X \to Y \ \forall y \in Y, \exists x \in X,\ y = f(x) \ .$$(Jackson 2017)

A bijective function is both injective and surjective.(Jackson 2017) A bijection, also known as a one-to-one correspondence, defines an exact correspondence between its domain and codomain.(O'Leary 2015) This type of correspondence implies invertibility, it can be "undone",(Jackson 2017 p. 514) for any element a we can apply $f(\ )$ to get $f(a)$ and then apply the inverse to recover a.(Jackson 2017) Let function $f : X \to Y$ be invertible, then there is a function $f^{-1} : Y \to X$ by which $f^{-1}(f(a)) = a$.(O'Leary 2015) A function is invertible if and only if it is a bijection.(O'Leary 2015) This will be of much relevance in later sections, Table 8 categorizes these behaviours.

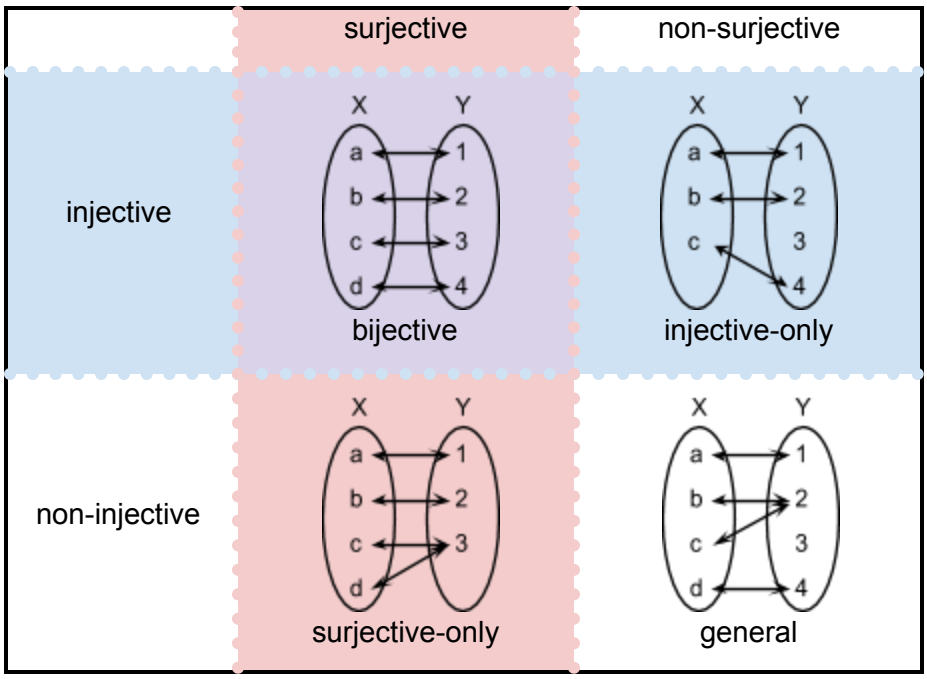

Table 8

A function's domain is composed of a number of sets $S_1 \times ... \times S_a$, equal to the operation's arity a.(Cohn 1981) The arity is a fixed non-negative integer characterizing the number of operands for which the operation is defined to produce output.(Cohn 1981) Commonly studied operations are binary operations, of arity 2, such as addition or multiplication, and unary operations, of arity 1, such as additive inverse or multiplicative inverse.(Cohn 1981) Application of the term "arity" is rare, examples such as "the addition operation has arity 2" are quite unusual, convention suggests we say "addition is a binary operation".

A general binary operation $\circ$, on a set S, is a function f which takes $s, t \in S$ and produces $f(s, t) \in S$, denoted $f : S \times S \rightarrow S$.(Jackson 2017) A binary operation $\circ$ joins operands x and y, $x \circ y$.(Jackson 2017) Furthermore, a binary operation $\circ$, is said to be "on" a set S if it's two domains and codomain are S, $\circ : S \times S \rightarrow S$.(Jackson 2017) In literature, binary operations are usually denoted by associated infix operators, rather than $f(s, t)$. Infix operation is indicated by the sequencing of operators between effected operands such as $s + t$, $s * t$, or st.(Sedgewick Wayne 2011)

We now define terminology for the two traditional arithmetic operations and their inverses:
The result of the addition of a and b, denoted a + b, is called the sum of a and b.
The result of the subtraction of a and b, denoted a - b, is called the difference of a and b.
The result of the multiplication of a and b, denoted ab or a·b, is called the product of a and b.
The result of the division of a and b, denoted a÷b or a/b, is called the quotient of a and b.

Section 4.3 : Properties of Set Operations

Throughout this document it will be necessary to show an operation modeled in this way has certain structural properties. Addition, subtraction, multiplication, and division on the integers $\mathbb{Z}$, are well-known binary operations. We will show that these operations adhere to certain fundamental properties.

For example, we can differentiate a general binary function, and a binary operation defined on a set. A function is formally defined as a relation between sets that associates each domain member with a codomain member, $f : X \to Y \ \forall y \in Y,\ \exists x \in X,\ f(x) = y$. A function f is then said to be binary if the domain is of arity 2, $f : X \times Y \to Z,\ \ \forall z \in Z,\ \exists x \in X \ and \ y \in Y,\ f(x, y) = z$. We have seen that a binary function where sets $X = Y = Z$, is known as a binary operation. This difference is formally defined as the property of closure.[(Cohn 1981)] A set is closed under a given operation if the resultant is a member of the set from which its operands were evaluated. The simplest example is the trivial set {0}, containing only zero. We find that the trivial set {0} is closed under the operations of addition, subtraction and multiplication, $0 + 0 = 0\ 0 - 0 = 0\ and\ 0 \times 0 = 0$.

An example that illustrates this difference that also has broader implications to the material to come, It can be seen that division of integers is not a closed binary operation, $\forall a \in \mathbb{Z},\ \exists b \in \mathbb{Z} \mid a/b \notin \mathbb{Z}$, for an integer a , there exists integer b , such that division produces a member of the codomain that is not an integer and therefore not a member of the domain. In later sections we will see how many of the common operations in both algebra and formal logic have interesting properties associated with the relationships they specify. These properties include closure, associativity, comutivity, distributivity, and the existence of identity and inverse elements.

Section 5 : Number Theory

5.1 : Introduction to Modular Arithmetic

This section facilitates the understanding of cryptographic algorithms through the examination of relevant mathematical operations. Modern cryptographic primitives implemented in both symmetric and asymmetric ciphers are based on arithmetic within a finite number of elements. Not only is modular arithmetic a common way of performing arithmetic in a finite set of integers, it is the method implemented by the AES.(Paar & Pelzi 2009) As such, understanding modular arithmetic and its application is of fundamental importance in the context of this report as well as in the greater scope of modern cryptographic study and practice.

The mechanism of modular arithmetic is essentially identical to the method by which we perform clock arithmetic.(Paar & Pelzi 2009) For example, moving between the 24-hour and 12-hour clock systems is a familiar conversion. One takes the value in the 24-hour clock system and reduces the hour by 12. 13:00 in the 24-hour clock system is 1:00 in the 12-hour clock system, this relationship of equality is captured by the operation 13 modulo 12 = 1.

We examine the 12 hour system again, an example of a finite set of integers from everyday life:

Consider the hours on a typical wall clock. If you keep adding one hour, you obtain:

1h,2h,3h,...,11h,12h,1h,2h,3h,...,11h,12h,1h,2h,3h,...

Even though we keep adding one hour, we never leave the set.

What are the practical implications? Imagine counting hours by the week instead of the modulus 12 system used in practice. We could no longer work a 9-5 job, instead we now hold down a 9-17, 33-41, 57-65, 81-89, 105-113 job, remembering that we have hours 129-137 and 153-161 off for the weekend. As we can see, arithmetic with a finite set has practical application. By modular arithmetic, we see what is known as a congruence, $9 \equiv 33 \equiv 57 \equiv 81 \equiv 105$ each represents 9AM. This is due to the divisibility of the number of hours in a week by the number of hours represented by the wall clock: 168/12 = 14

Now that we have some experience with the system by which we have a general way of dealing with arithmetic in finite sets, we can formally examine and define modular arithmetic. We shall first explore a common arithmetic operation, division, which has several properties that relate to the operation of modular arithmetic. We will then cover modular arithmetic operations themselves, followed by the operations of the AES which use modular arithmetic as a component.

5.2 : Divisibility

For two integers a and b, $a, b \in \mathbb{Z}$, where $a \neq 0$, $a|b$ means $\exists c \in \mathbb{Z}$ such that $b = ac$.(Stallings 2017) We then say a divides b indicating that $\frac{b}{a} \in \mathbb{Z}$ and a is said to be a divisor or factor of b.(Stallings 2017) For two integers a and b, $a, b \in \mathbb{Z}$, where $a \neq 0$ and $\frac{b}{a} \notin \mathbb{Z}$, then a does not divide b, we write $a \nmid b$.(Stallings 2017)

Examples (i) −3|18, since 18 = (−3)(−6). (ii) 173|0, since 0 = (173)(0).

Properties of divisibility $\forall a, b, c, x, y \in \mathbb{Z}$:(Stallings 2017)

- If a|1, then a = ±1
- If a|b and b|a, then a = ±b
- If a|b, then a|bc
- b|0, ∀b ≠ 0
- If a|b and b|c, then a|c
- If a|b and a|c, then a|(bx + cy)

An integer c, $c \in \mathbb{Z}$, is a common divisor of a and b if $c|a$ and $c|b$.(Stallings 2017) A non-negative integer d, $d \in \mathbb{Z}^+$, is the greatest common divisor (gcd) of two integers a and b, $a, b \in \mathbb{Z}$, denoted $d = gcd(a, b)$, if d is a common divisor of a and b; and whenever $c|a$ and $c|b$, then $c|d$.(Stallings 2017) Equivalently, $gcd(a, b)$ is the largest positive integer that divides both a and b.(Stallings 2017) As we shall see, numbers that are related by a gcd of 1, $gcd(a, b) = 1$, have great importance to the implementation of AES as well as many fields related to number theory.(Paar & Pelzi 2009) For $a, b \in \mathbb{Z}$, if $gcd(a, b) = 1$ then a and b are said to be relatively prime or coprime.(Stallings 2017)

Euler's totient function is defined as the number of positive integers less than n that are relatively prime to n.(Stallings 2017 p. 65) The number of integers $x = |k|$, where $k = \{ y \mid gcd(y, n) = 1, 1 \leq y \leq n \}$. The integer x is equal to the cardinality of k, $|k|$, where k is the set of integers y for which the greatest common divisor is equal to 1, $gcd(n, y) = 1$.(Stallings 2017) This function is denoted using the Greek letter ϕ, it is also known as Euler's phi function.(Paar & Pelzi 2009 p. 165) We will represent the set of integers coprime to n as $\phi(n)$.

An integer p, $p \in \mathbb{Z}$, where p is greater than 1, $p > 1$, is prime if, for all positive integers n, $n \in \mathbb{Z}^+$, less than p, $1 < n < p$ the only positive divisors are 1 and p, $gcd(p, n) = 1$.(Stallings 2017) Otherwise, p is composite.(Stallings 2017) $\mathbb{P}$ is the set of all primes {2, 3, 5, 7, 11, ...}.(Stallings 2017) If p is prime $p \in \mathbb{P}$ and $p|ab$, then either $p|a$ or $p|b$.(Stallings 2017) For n∈ℙ, ϕ(n) = {0, 1, …, (n-1)} as prime numbers are coprime to all other integers by definition.(Stallings 2017)

Divisibility Theorem[(Stallings 2017)]

If $a, d \in \mathbb{Z}$ and $d \neq 0$, then unique integers q and r exist, with $0 \leq r < d$, such that $a = d \cdot q + r$.

- d is called the divisor
- a is called the dividend
- q is called the quotient
- r is called the remainder

By example, $a = 17$, $d = 3$, we find $q = 5$ and $r = 2$ so that $17 = 3 * 5 + 2$.

If $r = 0$ for integers a and d, the division theorem reveals a relationship. We then say d divides a or that a is divisible by d. For any value of d, we could create a set of integers for which their remainder $r = 0$. This set would then be the infinite set of multiples of d, an abstraction with which most are familiar and can envision for a given integer d.

For example is $d = 5$, is the set $\{..., -10, -5, 0, 5, 10, 15...\}$.

5.3 : Congruence Relation

We now think about value sets which share the same remainder when divided by d, where the remainder $r \neq 0$. For any given divisor d, we have a remainer range of $0 \leq r < d$, and one such infinite value set for each member of this range, where all member values are said to be congruent to their entire set.(Stallings 2017) This establishes a congruence relation on the integers $\mathbb{Z}$, represented by the modulus operator.(Stallings 2017)

For a positive integer m, two integers a and b are said to be congruent modulo m, if a and b have the same remainder when divided by m.(Stallings 2017)

For $a, b, p, q \in \mathbb{Z}$ and $m \in \mathbb{Z}^+$, a is congruent to b modulo m iff $a - (q \cdot m) = r = b - (p \cdot m)$

Equivalently, if the difference of a and b is divisible by m, i.e. (a − b) is an integer multiple of m, then a and b are congruent modulo m.(Stallings 2017)

$\forall a, b \in \mathbb{Z}$ and $m \in \mathbb{Z}^+$, a is congruent to b modulo m iff $m \mid (a - b)$

i.e. $(\exists k \in \mathbb{Z},\ a - b = km)$

$\forall a, m, r \in \mathbb{Z}$, $\exists q \in \mathbb{Z}$ and $0 \leq r < m$, such that m divides $a - r$

as m divides $a - r$, we write $a \equiv r \ (mod\ m)$ and m is called the modulus.(Stallings 2017) We may then denote the remainder r, $a \bmod m = r$.(Stallings 2017)

The expression $a \equiv b \ (mod\ m)$ is called a congruence relation, read: a is congruent to b modulo m.(Paar & Pelzi 2009) Let it be known that some literature uses = instead of ≡ to denote congruence. We must also be aware of when the parentheses enclosing $(mod\ m)$ are omitted, as this denotes the modulo operation $a = b \bmod m$, expressing $0 \leq a < m$.(Stallings 2017) The integer m is known as the modulus of the congruence.(Stallings 2017)

We may rewrite a congruence relation $(mod\ m)\ a \equiv km + b$, explicitly showing its relationship with division.(Stallings 2017) We can see that b is not necessarily the remainder, b can be any member of a set of infinite values of a given congruence relation, as previously stated. Generally,

$a \equiv b \ mod \ m$ declares that a and b have the same remainder when divided by m.[(Stallings 2017)] That is, $a = pm + r$ and $b = qm + r$ where $0 \leq r < m$ is the common remainder.[(Stallings 2017)] By taking the difference of these expressions and setting $k = p - q$ we find: $a - b = km$.[(Stallings 2017)]

Example

$38 \equiv 14 \ (mod \ 12)$ because $38 - 14 = 24$, which is a multiple of 12, or, equivalently, because both 38 and 14 have the same remainder 2 when divided by 12.

5.4 : Modular Arithmetic

The (mod n) operator maps all integers into the set of integers {0, 1, … , (n - 1)}.[Stallings 2017] We can perform arithmetic operations in a finite set via modular arithmetic.

Properties of Modular arithmetic:[Stallings 2017]

[(a mod n) + (b mod n)] mod n = (a + b) mod n

[(a mod n) - (b mod n)] mod n = (a - b) mod n

[(a mod n) * (b mod n)] mod n = (a * b) mod n

If a and b are congruent modulo m, then they are of equivalent value and produce equivalent results as operands during modular arithmetic under m.[Stallings 2017] While any continuous sequence of m numbers would technically be of equal value and be able to represent the set of modular arithmetic under m, convention and simplicity dictate the first m natural numbers, {0, 1, 2, ..., (m-1)}. All other numbers are then congruent to and can be expressed as a member of this set for the purposes of modular arithmetic. Congruence $(\equiv)$, like equality $(=)$, is a type of equivalence relation.[Jackson 2017] We define the set $\mathbb{Z}_n$ to be the set of potential integer remainders modulo n, $\mathbb{Z}_n$ = {0, 1, ..., (m − 1)}. This is also the set produced by operator (mod m). |S| denotes cardinality, and is equal to the number of elements in the set S, thus $|\mathbb{Z}_n| = n$.[Jackson 2017]

$\mathbb{Z}_n$ can be equipped with an additive and multiplicative binary operations, + and *.[Stallings 2017] Addition and multiplication in $\mathbb{Z}_n$ function similar to addition and multiplication as defined over the integers $\mathbb{Z}$, except that the results are reduced modulo n.

Suppose we must compute 14*19 in $\mathbb{Z}_{15}$, given that $14 * 19 = 266$ in $\mathbb{Z}$.

We reduce 266 modulo 15: $266 = 15 * 17 + 11$, so $266 \; mod \; 15 \equiv 11$, and hence $14 * 19 = 11$ in $\mathbb{Z}_{15}$.

We consider the set $\mathbb{Z}_9$ = {0,1,2,3,4,5,6,7,8} in which arithmetic is seemingly regular for results smaller than 9: $2 * 3 = 6 \quad 4 + 4 = 8$. But what about 8+4? As with the previous example, we perform integer arithmetic then find the remainder with respect to the modulus, 9. As 8+4 = 12, and (12-3)/9 = 1, the remainder 3 is said to be congruent to 8 + 4 modulo 9.

Example: For every given modulus m and integer a, there are (infinitely) many valid remainders.(Paar & Pelzi 2009) Assuming we would like to reduce 13 modulo 8 there are infinitely many congruent results:

$13 \equiv 5 \pmod 8$, 5 is a valid remainder since $8|(13-5)$

$13 \equiv 21 \pmod 8$, 21 is a valid remainder since $8|(21-5)$

$13 \equiv -3 \pmod 8$, −3 is a valid remainder since $8|(-3-5)$

There is a system behind this behavior. The set of numbers {...,−19,−11,−3,5,13,21,29,...}, all of which are congruent, form what is called an equivalence class.

There are eight other equivalence classes for the modulus 9:

$0 \pmod 9$ {...,−18,−9, 0, 9,18,...}

$1 \pmod 9$ {...,−17,−8, 1, 10,19,...}

…

$8 \pmod 9$ {...,−10,−1, 8, 17,26,...}

Generally, each modulus has m equivalence classes {0, …, m-1}. By the 9-5 example, it is shown how a timekeeping system becomes less practical as the size of the representative set grows, i.e. the longer it takes to return to 0. Difficulty is experienced upon conversion between the set of hours on a wall clock 12, and the set of hours in a week, 168. Generally, numbers become increasingly difficult to compute as their digits grow. The modulus operator can assist us in reducing intermediate results to simplify future calculations by reducing the size of the numbers to be operated upon.(Paar & Pelzi 2009) Ultimately, for a given modulus m, it does not matter which equivalence class element we choose for computation.(Paar & Pelzi 2009) We show this using $3^8 \ (mod\ 7)$ and two modular exponentiation methods.

The first approach is a straight forward eight multiplication operations where by we obtain a large intermediate result of 6561 $3^8 = 6561 \equiv 2 \ (mod\ 7)$, before a modular reduction that ensures our final result is no larger than 6, since $6561 = 937 \times 7 + 2$.(Paar & Pelzi 2009)

For the second approach we perform two partial exponentiations: $3^8 = 3^4 * 3^4 = 81 * 81$ then replace the intermediate results 81 with a member of its equivalence class, the smallest positive

member modulo 7 in the class is 4 $(81 = 11 * 7 + 4)$,
$3^8 = 3^4 * 3^4 = 81 * 81 = 4 * 4 = 16 \equiv 2 \ (mod\ 7)$.(Paar & Pelzi 2009)

For the first method, dividing 6561 by 7 is mentally challenging, and an inefficient use of computational resources. The second method's operands never become larger than two digits. Generally, for computations with a fixed modulus, those common in cryptography, it is computationally advantageous to apply the modulo reduction to keep operands, and their results, small as we are free to choose the class element that results in the easiest computation.(Paar & Pelzi 2009) This property of equivalent classes has major practical implications as operations in many practical public-key schemes rely on exponentiation of the form $x^e \ mod\ m$, where x,e,m can be as great as 2048 bit integers.(Paar & Pelzi 2009)

5.5 : Exclusive OR

Pre-modern cipher systems were operated by hand, making them tedious and prone to human error. With the development of rotor cipher machines during World War I[(Van Tilborg & Jajodia, 2011)] and the invention of computers during World War II[(Van Tilborg & Jajodia, 2011)], the application of automated cryptographic operations immensely more complex than the ancient hand computed methods were made possible. Due to cryptology's crucial role in the outcome of both world wars, the continuous development of electromechanical devices, and the invention of digital computers, the cryptographic methods and their applications have progressed in capability, accessibility, and prevalence. Modern applications implement encryption operations as software programs, rather than pencil and paper, they instead use binary data files to represent input plaintext and key material, as well as for recording of the output ciphertext before storage or transmission. This implementation method has been the most effective for the automation of data entry and processing, particularly for algorithmic application of cryptographic operations. Such as the mixing function used by the one time pad.

This mixing operation is the exclusive OR, a logical operation that outputs true only when inputs differ in truth value, i.e. one is true, the other false.[(Van Tilborg & Jajodia, 2011)] The Inclusive-or operation results true when either or both inputs are true.[(Van Tilborg & Jajodia, 2011)] We shall represent the exclusive-or function by XOR or $\oplus$. When using binary values for true (1) and false (0) exclusive or functions by comparing two bits, returning 1 if the two bits differ and 0 if they are the same.[(Paar & Pelzi 2009)] This is equivalent to addition modulo 2 or performing addition in base 2 without carrying, seen by Table 9 to the left.[(Paar & Pelzi 2009)] This operation, known as a bitwise XOR, is defined to function on equal length bit patterns by transforming corresponding bits.[(Paar & Pelzi 2009)] Example: 1110_2 XOR $1001_2 = 0111_2$

XOR		=
1	1	0
1	0	1
0	1	1
0	0	0

Table 9

The exclusive or operation has several uses:(Paar & Pelzi 2009)

- Determines the equality of two individual bits
- Discretionary bit toggling, alters a bit xored with 1 and retains a bit xored with 0.
- A series of XORs, $a \oplus b \oplus \ldots \oplus z$, is true for an odd number, false for an even number, of true values

The XOR operation plays a major role in modern cryptography and will be used many times in the remainder of this document.(Paar & Pelzi 2009) On its own, exclusive-or is used as a cryptographic mixing function.(Paar & Pelzi 2009) For a simple example of exclusive-or's relevance, we xor a pseudorandom bit stream r with a message bit stream m to produce an encrypted bit stream c , $c_i = r_i \oplus m_i$, where i is the ith bit of a given stream.(Paar & Pelzi 2009) To decrypt this mixing function and restore the original message bit stream m , we xor the previous pseudo-random bit stream r with the encrypted bit stream c $r_i \oplus c_i = r_i \oplus r_i \oplus m_i = 0 \oplus m_i = m_i$, where i is the ith bit of a given stream.(Paar & Pelzi 2009)

Why is the XOR operation used? Let us consider the encryption of the plaintext bit x_i , if $x_i = 0$ is dependent on the key bit k_i . The ciphertext value y_i , is $y_i = 0$ if k_i = 0 or $y_i = 1$ if k_i = 1. If the key bit k_i behaves perfectly randomly, i.e., it is unpredictable, it has exactly a 50% chance to have the value 0 or 1, then both possible ciphertexts also occur with a 50% likelihood. Likewise, for plaintext bit $x_i = 1$, depending on the value of the key stream bit k_i , there is equal chance that the ciphertext $y_i = 0$ if k_i = 1 or $y_i = 1$ if k_i = 0. By observing an output value, there is exactly a 50% chance of any input bit value.(Paar & Pelzi 2009) The XOR operation preserves randomness, meaning that a random bit XORed with a non-random bit will produce a random result.(Paar & Pelzi 2009) Exclusive-or is heavily used in block ciphers such as AES (Rijndael) and in block cipher implementation and modes of operation.

5.6 : Polynomial Arithmetic

We shall represent a polynomials as mathematical expressions of the form

$$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_0$$

The highest exponent of x is the polynomial's degree(Dong 2010), for example, the degree of $x^7 + 2x^4 + 6$ is 7. The values a_n, a_{n-1}, ... , a_0 are called coefficients.(Dong 2010)

We add and subtract polynomials by operations between their like terms, those with the same degree.

Polynomial Addition:(Dong 2010)

$(x^5+3x^3+4)+(6x^6+4x^3) = 6x^6 + x^5 + 7x^3 + 4$

$$\begin{array}{r} x^5 + 3x^3 + 4 \\ +\ 6x^6 + \quad +4x^3 \\ \hline 6x^6 + x^5 + 7x^3 + 4 \end{array}$$

Figure 7

Polynomial Subtraction:(Dong 2010)

$(x^5+3x^3+4)+(6x^6+4x^3) = -6x^6+x^5-x^3+4$

$$\begin{array}{r} x^5 + 3x^3 + 4 \\ -\quad 6x^6 + \quad +4x^3 \\ \hline -6x^6 + x^5 - 1x^3 + 4 \end{array}$$

Figure 8

We can also multiply and divide polynomials by recording how each term in the first polynomial effects each term in the second polynomial, then summing those intermediate results.

Polynomial Multiplication:(Dong 2010)

$(x^5 + 3x^3 + 4)*(6x^6 + 4x^3) = 6x^{11} + 18x^9 + 4x^8 + 36x^6 + 16x^3$

$$
\begin{array}{rr}
 & x^5 + 3x^3 + 4 \\
\times & 6x^6 + 4x^3 \\
\hline
 & 4x^8 + 12x^6 + 16x^3 \\
6x^{11} + 18x^9 & + 24x^6 \\
\hline
6x^{11} + 18x^9 + 4x^8 + 36x^6 + 16x^3 &
\end{array}
$$

Figure 9

Polynomial Division:(Dong 2010)

$(6x^{11} + 18x^9 + 4x^8 + 36x^6 + 16x^3) \div (x^5 + 3x^3 + 4) = 6x^6 + 4x^3$

$$
\begin{array}{r}
6x^6 + \quad 4x^3 \\
\hline
x^5 + 3x^3 + 4 \;\big)\, 6x^{11} + 18x^9 + 4x^8 + 36x^6 + 16x^3 \\
\underline{6x^{11} + 18x^9 + \quad 24x^6} \\
4x^8 + 12x^6 + 16x^3 \\
\underline{4x^8 + 12x^6 + 16x^3} \\
0
\end{array}
$$

Figure 10

A dividend not perfectly divisible by the offered divisor is represented by a polynomial remainder.(Dong 2010)

Polynomial Division With Remainder:(Dong 2010)

$(3x^6 + 7x^4 + 4x^3 + 5) \div (x^4 + 3x^3 + 4) = 3x^2 - 9x + 34$ with remainder $-98x^3 - 12x^2 + 26x - 131$

$$
\begin{array}{r}
3x^2 - 9x + 34 \\
x^4 + 3x^3 + 4 \overline{)\,3x^6 + \quad 7x^4 + 4x^3 \quad + 5} \\
\underline{3x^6 + 9x^5 \quad + 12x^2} \\
-9x^5 + 7x^4 + 4x^3 - 12x^2 \quad + 5 \\
\underline{-9x^5 - 27x^4 \quad - 36x} \\
34x^4 + 4x^3 - 12x^2 + 36x + 5 \\
\underline{34x^4 + 102x^3 + \quad 136} \\
-98x^3 - 12x^2 + 36x - 131
\end{array}
$$

Figure 11

5.7 : Matrix Multiplication

A matrix is denoted by bold capital letters, A, B, C as shown by Figure 12 below.(O'Leary 2015) We then indicate individual members of matrix A by index notation such that member a_{ij} then expresses the matrix entry on the ith row and jth column.(O'Leary 2015)

Matrix multiplication is then performed with A, an n×m matrix, and B, a m×p matrix, such that the matrix product C, C = AB, is an n×p matrix.(O'Leary 2015)

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} \mathbf{B} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mp} \end{pmatrix} \mathbf{C} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{np} \end{pmatrix}$$

Figure 12

Each product matrix entry c_{ij} is the result obtained by individually multiplying an entry of the ith row of A and the jth column of B, then summing these m products.(O'Leary 2015)

$$c_{ij} = a_{i1}b_{1j} + \ldots + a_{im}b_{mj} = \sum_{k=1}^{m} a_{ik}b_{kj} \quad \text{for i = 1, ..., n and j = 1, ..., p.}$$

$$c_{11} = a_{11}b_{11} + a_{12}b_{21} + \ldots + a_{1m}b_{m1}$$

$$c_{i1} = a_{i1}b_{11} + a_{i2}b_{21} + \ldots + a_{im}b_{m1}$$

$$c_{1j} = a_{11}b_{1j} + a_{12}b_{2j} + \ldots + a_{1m}b_{mj}$$

...

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \ldots + a_{im}b_{mj}$$

Thus the product AB is defined if and only if the number of columns in A equals the number of rows in B.(O'Leary 2015) Figure 13, below, depicts the method of matrix multiplication for two members of matrix C, the product of the two matrices denoted by the capital letters A and B in Figure 14, below. This diagram shows how each intersection in the product matrix corresponds to a combination between a row of A and a column of B.

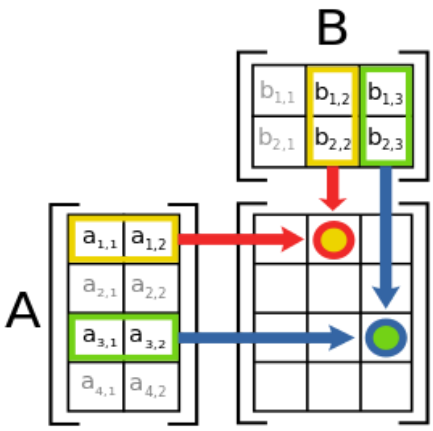

Figure 13

$$\underset{4\times 2 \text{ matrix}}{\begin{bmatrix} a_{11} & a_{12} \\ \cdot & \cdot \\ a_{31} & a_{32} \\ \cdot & \cdot \end{bmatrix}} \underset{2\times 3 \text{ matrix}}{\begin{bmatrix} \cdot & b_{12} & b_{13} \\ \cdot & b_{22} & b_{23} \end{bmatrix}} = \underset{4\times 3 \text{ matrix}}{\begin{bmatrix} \cdot & x_{12} & x_{13} \\ \cdot & \cdot & \cdot \\ \cdot & x_{32} & x_{33} \\ \cdot & \cdot & \cdot \end{bmatrix}}$$

Figure 14

The values indicated by Figures 13, 14 above are solved for:

$$x_{1\,2} = a_{1\,1}b_{1\,2} + a_{1\,2}b_{2\,2}$$
$$x_{3\,3} = a_{3\,1}b_{1\,3} + a_{3\,2}b_{2\,3}$$

Traditionally matrix entries are numbers, but they may be any kind of mathematical objects for which an addition and multiplication operations are associative, the addition is commutative, and the multiplication is distributive with respect to the addition.(O'Leary 2015)

Section 6 : Abstract Algebra

6.1 : Algebraic Structure

Now that we have some understanding of set theory, its notation, and the creation of binary operations, we are able to progress toward the necessary structures and operations of the AES. The mathematical constructs implemented and manipulated by AES are sets with defined structure. Through selective abstraction, mathematicians have defined algebraic structures now integral to both pure mathematics and the applied sciences. Almost all systems studied are sets, abstract algebra in particular, is based on set theory in one form or another.(Jackson 2017)

This section concerns those properties that define the algebraic structures which mathematically model the mechanisms of the AES. We shall see how the definition of a binary operation and the relation it creates, affects the set for which it is defined. We begin from the most basic structure, a general set for which a binary operation is defined, then progress through the individual properties necessitated by the operations of the AES and define the algebraic structures which result as they increase in complexity. This incremental addition of properties creates a hierarchy of algebraic structures. Groups, rings, and fields constitute the basic hierarchy of abstract algebraic objects and are required for the definition and understanding of the AES.(Sedgewick Wayne 2011) "The abstract formalization of the group axioms, detached as it is from the concrete nature of any particular group and its operation ... allows entities with highly diverse mathematical origins in abstract algebra and beyond to be handled in a flexible way while retaining their essential structural aspects. The ubiquity of groups in numerous areas within and outside mathematics makes them a central organizing principle of contemporary mathematics".(Bello Danjuma Simon 2018 p. 54) The approach adopted by this document is known as naïve set theory, by which we define set relations and focus on their functional properties without concern for those unusual circumstances under which exceptions occur.(Jackson 2017) In practice, the definitions and functionality outlined by this document do not involve such exceptions.(Jackson 2017)

An algebraic structure is a collection of finitary operations on a set S, also called an algebra.(Cohn 1981) The set S is then referred to as the underlying set.(Cohn 1981) A finitary operation is an operation of finite arity, that is an operation of limited arguments or operands.(Cohn 1981) An algebraic structure is denoted $(S, \circ)$ where S represents a set and ∘ represents a properly defined operation.(Cid Murphy Robshaw 2006) This document shall use the symbol '∘' to generalize any binary operation. We can represent more complex algebraic structures as ordered tuples: $(S, \circ_1, \circ_2, ..., \circ_n)$ where S is a set which has one or more binary operations $\circ_i$, $1 \leq i \leq n$.(Cid Murphy Robshaw 2006)

Closure

An algebraic structure (S,∘) is closed under ∘ iff $\forall(a,b) \in S \times S : a \circ b \in S$ (Paar & Pelzi 2009)

In general, we may state that S is closed under ∘, or that $(S, \circ)$ is closed, if the binary operation ∘, combines any two elements a,b ∈ S, by some mechanism, denoted $a \circ b$ or ab, such that their result is always a member of S.(Paar & Pelzi 2009) A set S, equipped with a binary operation ∘, is called a magma if (S, ∘) is closed, S ∘ S → S.(Jackson 2017) As the binary operations which define set structure are closed by definition, the magma will serve as our most basic example of an algebraic structure.(Jackson 2017)

Associativity

A binary operation ∘, on a set S, is associative iff $\forall a, b, c \in S\ (a \circ b) \circ c = a \circ (b \circ c)$ (Paar & Pelzi 2009)

A set S, equipped with a binary operation ∘, is called a semigroup if $(S, \circ)$ is closed and associative.(Jackson 2017)

Commutative

A binary operation ∘, on set S, is commutative iff $\forall a, b \in S\ a \circ b = b \circ a$ (Paar & Pelzi 2009)

If semigroup $(S, \circ)$ is commutative it is known as an abelian semigroup.(Jackson 2017)

Identity

An element e is the Identity element of set S, iff ∀a ∈ S ∃ e ∈ S e ∘ a = a ∘ e = a (Paar & Pelzi 2009)

The element e is said to be the identity element of, or neutral element with respect to, ∘.(Jackson 2017) An identity element, when combined with a set member, with respect to a binary operation, results in an identical value as that set member.(Paar & Pelzi 2009) The identity of addition is 0 and the identity of multiplication is 1.(Paar & Pelzi 2009) A set S, equipped with a binary operation ∘, is called a monoid if $(S, \circ)$ is closed, associative, and has an identity element e.(Jackson 2017) The monoid therefore is characterized by specification of the triple $(S, \circ, e)$.(Jackson 2017) We begin to see a pattern, a monoid is a semigroup with an identity element or a magma with associativity and identity.

6.2 : Groups

Invertible

Binary operation ∘ on set S with identity e is invertible iff $\forall a \in S \; \exists a^{-1}, \; a \circ a^{-1} = e = a^{-1} \circ a$ (Paar & Pelzi 2009)

An inverse element, with respect to a binary operation, can reverse, or invert the effect of combination with a corresponding element, this is known as inversion.(Paar & Pelzi 2009) Inversion manifests as negation under addition, and the reciprocal under multiplication.(Paar & Pelzi 2009) Extending this structural construct, a monoid in which each element has an inverse is a group. A set G, equipped with a binary operation ∘, is called a group if $(G, \circ)$ satisfies the conditions of closure, associativity, identity and invertibility.(Jackson 2017) These four conditions are called the group axioms:(Jackson 2017)

Group

(G, ∘) is a group iff if ∘ satisfies closure, associativity, identity and invertibility(Paar & Pelzi 2009)

Closure :	$\forall a, b \in G \mid a \circ b \in G$
Associativity :	$\forall a, b, c \in G \mid (a \circ b) \circ c = a \circ (b \circ c)$
Identity element :	$\exists e \in G, \forall a \in G \mid e \circ a = a \circ e = a$
Inverse element :	$\forall a \in G, \exists a^{-1} \in G \mid a \circ a^{-1} = a^{-1} \circ a = e$

The simplest possible construct that satisfies all of the group axioms is the trivial group, $G = \{0\}$, the set consisting of a single element.(Jackson 2017) There is only one possible binary operation that can be defined on this set, and only one ordered pair of elements upon which it can operate, with only a singular result that can be produced: $0 \circ 0 = 0$.(Jackson 2017) Brief verification shows that this operation satisfies all of the group axioms: 0 is the identity element, it's its own inverse, and the operation is shown to be closed and associative over the elements of the set.(Jackson 2017) One of the most familiar groups is the set of integers $\mathbb{Z}$, with the operation of arithmetic integer addition.(Jackson 2017) The following properties of arithmetic addition on the integers serve as an example for a set and operation which satisfy the group axioms:(Cid Murphy Robshaw 2006)

- $\forall a, b \in \mathbb{Z},\ (a+b) \in \mathbb{Z}$ For any two integers a and b, the sum $a+b$ is also an integer. Thus, the integers are closed under addition.
- $\forall a, b, c \in \mathbb{Z},\ (a+b)+c = a+(b+c)$ Adding a to b, and their result to c is equal to adding a to the result of $b+c$. Thus addition under the integers is associative.
- $\forall a \in \mathbb{Z},\ 0+a = a+0 = a$ For any integer a, adding it to zero or vice versa returns the same integer. Zero is thus the identity element of addition.
- $\forall a \in \mathbb{Z},\ \exists b \in \mathbb{Z},\ a+b = b+a = 0$ For every integer a, there is an integer b such that its addition to a results in 0. The integer b is then denoted $-a$, the inverse of a.

The integers, together with the operation +, form a mathematical object belonging to a broad class sharing similar structural characteristics. To appropriately understand and classify these structures, their definition is developed by the group axioms.

Any binary operation which preserves the above properties, with respect to some set G, represents a group over G. As it's properties dictate the set's algebraic structure, the associated operation is called the law of G.(Cohn 1981) Again, G is called the underlying set of the group $(G, \circ)$.(Cohn 1981) Often the group's underlying set G is used as a short name for the group $(G, \circ)$ as the operation can be discerned via context.(Jackson 2017) When this is the case, a usage of $(G, \circ)$ is instead denoted by G.(Jackson 2017)

We have required the existence of unique identity elements $e \in S$ and a unique inverse g^{-1} for each element $g \in S$.(Jackson 2017) We now prove that the identity element e of a group S is unique.

That is, for any other element $g \in S$ satisfying condition of Invertibility, there exists an identity element $e \in S$ such that $\forall g \in S,\ e \circ g = g \circ e = g$.(Jackson 2017)

Proof

Suppose $f \in S$ also satisfies the identity condition, that for any element $g \in S$, we have $f \circ g = g$ and $g \circ f = g$. In particular, $f \circ e = e$. But since e is also an identity, we have $f \circ e = f$ as well. So $f = e$.(Jackson 2017)

Any element g of a group G has a unique inverse g^{-1}. That is, for any other element g satisfying condition (IV), we have $g' = g^{-1}$ (Jackson 2017)

Proof

Suppose that g^{-1} and g' are two inverses for an element $g \in G$. Then $g^{-1} \circ g = g' \circ g = e$. By condition III we have $g \circ g^{-1} = e$ as well. Thus: $g^{-1} = e \circ g^{-1} = (g' \circ g) \circ g^{-1} = g \circ (g \circ g^{-1}) = g \circ e = g$ (Jackson 2017)

Hence the identity element and the inverse elements are shown to be unique.

The order of a group (G, ∘), often denoted |G|, is the cardinality of the set G.(Jackson 2017) If the order of (G, ∘) is finite, we say that G is a finite group.(Jackson 2017) Similarly, we say that an element g ∈ G has finite order if there exists a positive integer k such that g ∘ ... ∘ g = g^k = e, the identity element.(Jackson 2017) In this case, the least such integer k is called the order of g and is denoted by or |g|, thus the inverse element $g^{-1} = g^{(|g| - 1)}$.(Jackson 2017) Let g be an element of a group G with identity element e. Then the order of g, denoted |g|, is the smallest positive integer n such that g^n = e. If no such integer n exists, g is said to have infinite order.(Jackson 2017) For a finite group G, the order of any element divides the order of the group G.(Jackson 2017)

In the case of $\mathbb{Z}_4$, orders are given by

|1,3|=4, |2|=2, |0|= 1.

For $\mathbb{Z}_{12}$, the orders are

|1,5,7,11|= 12, |2,10|= 6, |3,9|= 4, |4,8|=3, |6|=2, |0|=1.

In both of these examples we see that the order of the identity element is 1, and furthermore that no other element apart from the identity has order 1, this is true in general. Let G be a group with identity element e. ∀g ∈ G, |g| = 1 if and only if g = e.(Jackson 2017)

Proof

The order of e is always 1, since 1 is the smallest positive integer n for which e^n = e. Conversely, if g^1 = e then g = e.(Jackson 2017)

The simplest, nontrivial, case is that of a group where all the nonidentity elements have order 2.(Jackson 2017) If we let G be a nontrivial group, all of whose elements apart from the identity have order 2.(Jackson 2017) Then G can be shown to be commutative.(Jackson 2017)

Proof

Let g,h ∈ G, by the hypothesis, g^2 = h^2 = e, and so $g = g^{-1}$ and $h = h^{-1}$. It then follows that, as required, $g \circ h = g^{-1} \circ h^{-1} = (h \circ g)^{-1} = h \circ g$.(Jackson 2017)

The group (G, ∘) is said to be an abelian group if ∘ is commutative.(Jackson 2017) Almost all groups implemented cryptographically are abelian, since the commutative property is often what makes them cryptographically interesting.(Paar & Pelzi 2009)

If a group, with group operation + and identity 0, is abelian, it is called an additive group.(Jackson 2017) An additive group operation is denoted by addition, f = g + h and 5 × g = g + g + g + g + g.(Jackson 2017) If a group, with group operation × and identity 1, is abelian it is called a multiplicative group.(Jackson 2017) A multiplicative group operation is denoted by multiplication, f = g × h and g^5 = g × g × g × g × g.(Jackson 2017) Roughly speaking, a group is a set with one operation and the corresponding inverse operation. If the operation is called addition denoted "+", the inverse operation is subtraction and the inverse element is denoted −a. If the operation is multiplication denoted "×", the inverse operation is division (or multiplication with the inverse element) and the inverse element is denoted a^{-1}.(Paar & Pelzi 2009)

Building on the operation of arithmetic addition on the integers ℤ, we consider this structure alongside others commonly used in mathematics, including those structures addressed by this text, to determine whether each example structure's binary operation and underlying set satisfy the group axioms. We begin with the algebraic properties of the groups (ℤ, +) and (ℤ, ×) in Table 10, below:

ℤ	Addition	Multiplication
Closure	a+b∈ℤ	a*b∈ℤ
Associativity	a+(b+c) = (a+b)+c	a*(b*c) = (a*b)*c
Commutativity	a+b = b+a	a*b = b*a
Identity	a+0 = a	a*1 = a
Inverse	a+(-a) =0	1*1 = 1, -1*(-1) = 1

Table 10

We can conclude that (Z, *), unlike (Z, +), does not satisfy the group axioms as most member's inverses are not integers, $\exists a \in \mathbb{Z}$, $a^{-1} \notin \mathbb{Z}$. Furthermore, addition of a to b is equivalent to addition of b to a, thus addition on the integers is commutative, so the structure $(\mathbb{Z}, +)$ is also an Abelian group. We now consider the group operations of modular addition and multiplication on the set of remainders modulo n, $\forall n \in \mathbb{Z}, \mathbb{Z}_n = \{0, 1, 2, ..., n-1\}$. The algebraic properties of $(\mathbb{Z}_n, +)$ and $(\mathbb{Z}_n, \times)$, are detailed by Table 11, below.

$\mathbb{Z}_n$	Addition modulo n	Multiplication modulo n
Closure	a+b ≡ c mod n, 0 ≤ c < n	a*b ≡ c mod n, 0 ≤ c < n
Associativity	a+(b+c) ≡ (a+b)+c mod n	a*(b*c) ≡ (a*b)*c mod n
Commutativity	a+b ≡ b+a mod n	a*b ≡ b*a mod n
Identity	a+0 ≡ a mod n	a*1 ≡ a mod n
Inverse	a+(n-a) ≡ 0 mod n	$\exists a^{-1} \in \phi(n)\ \forall a \in \phi(n)$

Table 11

We find that, as with their arithmetic counterparts, $(\mathbb{Z}_n, +)$ is a group and $(\mathbb{Z}_n, *)$ is not. $(\mathbb{Z}_n, +)$ is also an Abelian group with the neutral element 0 and inverse element $\forall a \in \mathbb{Z}_n,\ \exists -a \in \mathbb{Z}_n,$ $-a = (n-a)\ (mod\ n)\quad a + (-a) = 0\ mod\ n$. $(\mathbb{Z}_n, \times)$ does not form a group because not all elements a have an inverse such that $a \times a^{-1} = 1$ mod n. In fact, given that $0 \in \mathbb{Z}_n$ and 0 is not coprime to any number and is therefore non-invertible, $\forall n \in \mathbb{Z}$, $(\mathbb{Z}_n, \times)$ is not a group.

Finally, we consider the group operations of modular addition and multiplication on the set of remainders coprime to the modulus n. We will denote this set ϕ(n), Euler's totient function. The algebraic properties of (ϕ(n), +) and (ϕ(n), ×), are detailed by Table 12, below.

ϕ(n)	Addition modulo n	Multiplication modulo n
Closure	$\exists$ a+b ∉ ϕ(n)	a*b ≡ c mod n, c ∈ ϕ(n)
Associativity	a+(b+c) ≡ (a+b)+c mod n	a*(b*c) ≡ (a*b)*c mod n
Commutativity	a+b ≡ b+a mod n	a*b ≡ b*a mod n
Identity	No	a*1 ≡ a mod n
Inverse	$\exists a \in \phi(n) \mid -a \notin \phi(n)$	$\forall a \in \phi(n)\ \exists a^{-1} \in \phi(n)$

Table 12

Surprisingly, we now find that (ϕ(n), +) is not a group and (ϕ(n), *) is now an Abelian group.

For n =10, the order of ϕ(10), *) is 4 not 9 as ϕ(10) = {1, 3 , 7, 9}. We know that when n is a prime number, n∈ℙ, the order of (ϕ(n), *) is. |ϕ(n)|=n-1, e.g. ϕ(n) = {1, 2, ..., n-1}. More generally, the set of positive integers $\mathbb{Z}^{+}_{n}$ = {1, …, n-1 } under the operation of multiplication modulo n forms an abelian group if n is prime i.e. n∈ℙ.[Jackson 2017] We will use Z^{*}_{p} to denote this group, which is the multiplicative group of integers modulo prime p i.e. (ϕ(p), *) where p ∈ ℙ.[Jackson 2017]

6.3 : Cyclic Group

An abelian group is called cyclic if it contains at least one member which all others may be expressed with respect to.(Paar & Pelzi 2009) A group member of this type is denoted $g \in G$, and is called the generator of the group.(Paar & Pelzi 2009) Every group member is obtained by repeated application of the group operation with g or it's additive inverse $-g$.(Paar & Pelzi 2009)

Consider the integers under addition ($\mathbb{Z}$, +) we will show that it possesses a generator $g = 1$. A given positive integer is obtained by repeated application of the group operation with g, e.g. $4 = 1 + 1 + 1 + 1$. While 0 or a given negative integer is obtained by repeated application of the group operation with the additive inverse $-g$, e.g. -4 = 1+(-1)+(-1)+(-1)+(-1)+(-1).

Generally what the existence of a group generator allows is the ability ∀a∈G to be expressed as either a multiple of g, $kg = a,\ \exists g \in G\ and\ \exists k \in \mathbb{Z},$ for groups with additive notation or as an exponent of g, $g^k = a,\ \exists g \in G\ and\ \exists k \in \mathbb{Z},$ for groups with multiplicative notation. A generator g, of a cyclic group G, may be denoted by G = ⟨g⟩.(Jackson 2017) By the example above $\mathbb{Z}$ =⟨1⟩

If we let $+:\ \mathbb{Z}_n \circ \mathbb{Z}_n \rightarrow \mathbb{Z}_n$ be addition modulo n on the set $\mathbb{Z}_n$ by which we define the sum of any two a,b ∈ $\mathbb{Z}_n$ to be the remainder r, of the sum of $(a + b)$ divided by the modulos n represented $a + b = r(mod\ n),\ r \in \mathbb{Z}_n$.(Jackson 2017) This operation defines the cyclic group $\mathbb{Z}_n$ of order n, $|\mathbb{Z}_n| = n$.(Jackson 2017) The operation defined above can be represented by $\mathbb{Z}_{12}$ as shown by Figure 15, below. The elements of $\mathbb{Z}_{12}$, or any sized cyclic field, may be conceptualized "… geometrically as n equally spaced points around the circumference of a circle".(Jackson 2017 p. 7) We then resolve (a + b) by starting at point a then moving b positions clockwise.(Jackson 2017) By example, 5 + 9 = 14 ≡ 2 (mod 12).(Jackson 2017)

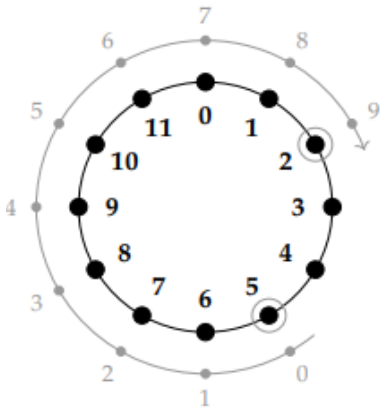

Figure 15

Beyond the fact that a cyclic group is generated by a single element, we derive a few general results about the order of these generators. Suppose G = ⟨g⟩ is a cyclic group. If G is an infinite cyclic group, by definition g can't have finite order, as G could then only contain finite members.(Jackson 2017) Therefore if G is an infinite cyclic group |G| = ∞, g has infinite order |g| = ∞.(Jackson 2017) If G is a finite cyclic group with g of finite order |g|=n, G contains exactly n members and had order |G|=n.(Jackson 2017)

Proof

For a generator of finite order |g|=n where n∈ℕ, $g^k = g^{k+n} = g^{k+pn},\ \forall k, p \in \mathbb{Z}$ (Jackson 2017)

This follows from g^k = e iff n|k, thus k is a multiple of n (k mod n = 0), hence G must contain at least n elements.(Jackson 2017) As well G = ⟨g⟩ = {g^k | k ∈ ℤ} = {g,g^2, ..., g^(|g|-1), e} contains at most n elements. Therefore G is a finite cyclic group |G| = n, if g is of finite order |g| = n.(Jackson 2017)

How do we know a given member g ∈ G generates a group without individually confirming each member's expression in terms of g? Let k ∈ $\mathbb{Z}_n$, k is a generator for $\mathbb{Z}_n$ iff $gcd(k,\ n) = 1$.(Jackson 2017)

Proof

k generates $\mathbb{Z}_n$ iff it has order n, $|k| = n$, as the smallest positive integer m such that n|(mk) is n itself, therefore k and n must be coprime i.e. $gcd(k,\ n) = 1$ (Jackson 2017)

As such, any integer k∈$\mathbb{Z}_n$, that is coprime to n can generate the finite cyclic group $\mathbb{Z}_n$,(Jackson 2017) and the number of $\mathbb{Z}_n$ generators may be denoted by ϕ(n), Euler's totient function from Section 5.2.(Paar & Pelzi 2009)

6.4 : Isomorphism

As we are interested in understanding those properties which define various algebraic structures, it is of great utility to have a method by which we could determine whether two given groups are equivalent.(Jackson 2017) An isomorphism is an equivalence between two structures, structures that share an isomorphism are then said to be isomorphic.(Cid Murphy Robshaw 2006) Isomorphic structures are those with equivalent algebraic properties.(Cid Murphy Robshaw 2006) By definition, groups with equivalent properties are fundamentally the same algebraic structure, technically indistinguishable.(Cid Murphy Robshaw 2006) We establish the existence of an isomorphism via bijection, a function of one-to-one correspondence with respect to the group operations.(Jackson 2017) More generally, given two structurally equivalent groups $(G,\circ)$ and $(H,\bullet)$, we want a bijection $f: G \to H$ such that the product in H of the images of any two elements of G is the same as the image of their product in G.(Jackson 2017)

Two groups $G = (G,\circ)$ and $H = (H,\bullet)$ are isomorphic, denoted $G \cong H$, if there exists a bijective function, which we will call an isomorphism, $f: G \to H$, such that ∀u,v∈G, $f(u \circ v) = f(u) \bullet f(v)$.(Jackson 2017)

For example, if we let p∈ℙ and $\mathbb{Z}_{p-1}$ be the group generated additively by 1 ∈ $\mathbb{Z}_{p-1}$, and let $\mathbb{Z}^*_p$ be the group generated multiplicatively by some g ∈ $\mathbb{Z}^*_p$, then $mg \to g^m$, multiplication in $\mathbb{Z}_{p-1}$ is equal to exponentiation in $\mathbb{Z}^*_p$ and defines an isomorphism between them, thus these groups are isomorphic.(Cid Murphy Robshaw 2006)

When we realize such relationships exist, the properties of a given isomorphism allow for alternative representation of some underlying set, a common technique in the study of algebraic structures.(Cid Murphy Robshaw 2006) Isomorphisms have found use solving problems that had been intractable otherwise.(Cid Murphy Robshaw 2006)

6.5 : Ring

Distributive

A binary operation ∘, on a set S, is distributive iff $\forall a, b, c \in R\ (a + b) \circ c\ =\ a \circ c + b \circ c$ (Paar & Pelzi 2009)

We carefully studied the set of integers $\mathbb{Z}$, giving particular attention to its additive structure.

Through abstraction of arithmetic addition on the set of integers we explored some properties associated with binary operations; closure, associativity, commutativity, and the existence of an identity and inverse elements. This led us to formulate the definition of a group, (G, ∘) which encompassed these axioms. We now define a structure of greater complexity, which defines a relationship between two such operations and their underlying set. Having established an understanding of the modulus operator in Section 5.4, we are now ready to define a structure that is based on modular arithmetic. The definition for the algebraic structure modeled by the set of integers $\mathbb{Z}_n$ under the operations of modular arithmetic addition and multiplication. We are able to generalize the relationship shared by the arithmetic operations of addition and multiplication via the algebraic structure called a Ring.(Jackson 2017) Through the application of rings, theorems derived in the context of arithmetic are found to be applicable to various mathematical objects such as polynomials and matrices, and vice versa.(Jackson 2017) A ring consists of a set R with two binary operations defined on its elements.(Jackson 2017) A given ring R, and it's operations which, we will denote '+' and '×', are represented by the triple (R, +, ×).(Jackson 2017) To qualify as a ring, an underlying set and its operations must satisfy the following properties, known as the ring axioms:(Jackson 2017)

Ring

(R, +, ×) is a ring iff (R, +) is an Abelian group, (R, ×) is a monoid, and × distributes over +. (Paar & Pelzi 2009)

1. (R, +) is an Abelian group:(Jackson 2017)

+ is associative: $\forall a, b, c \in R\ (a + b) + c\ =\ a + (b + c)$

+ is commutative: $\forall a, b \in R\ a + b\ =\ b + a$

additive identity: $\forall a \in R\ \exists 0 \in R\ a + 0\ =\ a$

additive inverse: $\forall a \in R\ \exists \text{-}a \in R\ such\ that\ a + (\text{-}a) = 0$

2. (R, ×) is a monoid under multiplication:[Jackson 2017]

× is associative: $\forall a, b, c \in R\ (a \times b) \times c = a \times (b \times c)$

multiplicative identity: $\forall a \in R\ \exists 0 \in R\ a \times 1 = a$

3. Multiplication is distributive with respect to addition:[Jackson 2017]

∀a,b,c ∈ R a×(b+c) = (a×b)+(a×c) (left)

∀a,b,c ∈ R (b+c)×a = (b×a)+(c×a) (right)

We see that the additive operation is required to be commutative, while the multiplicative is not: ∀a,b∈R a×b ≠ b×a. An example of a ring by which we might see this phenomenon is the set of n×n or square matrices with the operations matrix addition and matrix multiplication.[Jackson 2017] This set is a ring that is not commutative for $n > 1$.[Jackson 2017] Rings for which both operations satisfy the property of commutativity are known as commutative rings.[Jackson 2017] This new structure definition may seem complex, but these axioms characterize the sets most common in mathematics, for example the infinite commutative ring (ℤ, +, ∘): the set of integers, with the operations of arithmetic addition and multiplication.[Jackson 2017] For the majority of cryptologic practice and specifically during the operation of the AES, we consider the properties and mechanisms of finite rings, like the commutative ring of integers modulo N, $\mathbb{Z}/_N\mathbb{Z}$.[Smart 2016] As we have shown, $\mathbb{Z}/_N\mathbb{Z}$ is an abelian group when the group law is modular addition, we will now see that the group $\mathbb{Z}/_N\mathbb{Z}$ is also a ring if we consider the relationship between modular addition and multiplication.[Smart 2016]

Ring $\mathbb{Z}_n$ is defined as the set $\mathbb{Z}_n$ = {0, 1, 2, ... n − 1} and two operations:

Addition a+b ≡ c mod n ∀a,b,c ∈ $\mathbb{Z}_n$

Multiplication a×b ≡ c mod n ∀a,b,c ∈ $\mathbb{Z}_n$

Ring $\mathbb{Z}_n$ properties:

(+) identity	∀a ∈ $\mathbb{Z}_n$	a + 0 = a mod n
(+) inverse	∀a ∈ $\mathbb{Z}_n$	a + (−a) ≡ 0 mod n
	-a = (-a + m) mod m as 0 ≤ a<m.	
(+) is closed	∀a,b,c ∈ $\mathbb{Z}_n$	a + b ≡ c mod n
(+) is associative	∀a,b,c ∈ $\mathbb{Z}_n$	(a + b) + c = a + (b + c)
(+) is commutative	∀a,b ∈ $\mathbb{Z}_n$	a + b = b + a

(×) identity	$\forall a \in \mathbb{Z}_n$	$a \times 1 = a \bmod n$
(×) inverse	$\forall a \in \mathbb{Z}_n$	$a \times a^{-1} \equiv 1 \bmod n$
	$a^{-1} \in \mathbb{Z}_n$ if a,n are coprime gcd(a,n) = 1	
(×) is closed	$\forall a,b,c \in \mathbb{Z}_n$	$a \times b \equiv c \bmod n$
(×) is associative	$\forall a,b,c \in \mathbb{Z}_n$	$(a \times b) \times c = a \times (b \times c)$
(×) is commutative	$\forall a,b \in \mathbb{Z}_n$	$a \times b = b \times a$

We shall consider the integer ring of order 7 and compute addition and multiplication for 3 and 5 in $\mathbb{Z}_7$. For the rig $\mathbb{Z}_7$, we let modulus n = 7, and the ring members are represented by $\mathbb{Z}_7$ = {0,1,2,3,4,5,6}.(Paar & Pelzi 2009)

$$3+5 = 8 \equiv 1 \bmod 7 \qquad 3 \times 5 = 15 \equiv 1 \bmod 7$$

We will now highlight the following property of rings under modular arithmetic addition and multiplication the multiplicative inverse is not guaranteed to exists for all elements $e \in \mathbb{Z}_n$.(Paar & Pelzi 2009) If an inverse exists for a given element a, we are able to apply the binary operation of division and receive a defined result $\div: b \div a \equiv b \times a^{-1} (mod\ n)$.(Paar & Pelzi 2009) A convenient method by which to determine whether the inverse for a given element a exists or not is to determine the greatest common divisor of a and the integer modulus n, $gcd(a, n)$. An element $a \in \mathbb{Z}$ has a multiplicative inverse a^{-1} iff $gcd(a, n) = 1$.(Paar & Pelzi 2009) This means, if we determine that the largest integer that divides both numbers a and n is 1, a^{-1} exists. Methods for finding the $gcd(a, n)$ usually employ the Euclidean algorithm.(Paar & Pelzi 2009) For example to determine the existence of the multiplicative inverse of 15 in $\mathbb{Z}_{26}$, we see that $gcd(15, 26) = 1$, as such the inverse must exist. Conversely, since gcd(14,26) = 2 ≠ 1 the multiplicative inverse of 14 is not defined for $\mathbb{Z}_{26}$.

In general, we are able to classify the ring $\mathbb{Z}$n as the set of elements in which we can add, subtract, multiply, and sometimes divide.(Paar & Pelzi 2009) However, when we wish to ensure that division is possible for some $a,b \in \mathbb{Z}_n$, a/b = x (mod n) we must first deduce whether a multiplicative inverse modulo n exists for a given element a, as division is the inverse operation of multiplication.(Smart 2016) The multiplicative inverse of a modulo n is a ring member c such that $a \circ c = c \circ a = 1 \pmod n$.(Smart 2016) This value is then the inverse of a, denoted a^{-1}, and provides c for the equation $a \circ c = 1 \pmod n$.(Smart 2016) As we have seen, a^{-1} only exists iff $gcd(a, n) = 1$ i.e. when a and n are coprime. Of particular interest, then, is when n is a prime $n \in \mathbb{P}$, since then $\forall a \in \mathbb{Z}/p\mathbb{Z}\ gcd(a, p) = 1$ and we obtain a solution to $a \circ x = 1 \pmod p$ for all $a \neq 0$.(Smart 2016) Hence $\mathbb{Z}/p\mathbb{Z}$ makes the operation of division possible because $\forall a \in \mathbb{Z}/p\mathbb{Z}$, $\exists a^{-1} \in \mathbb{Z}/p\mathbb{Z}$,where $a \neq 0$.(Smart 2016) A commutative ring such that every nonzero element has a multiplicative inverse defines the algebraic structure known as a field.(Smart 2016)

6.6 : Fields

We have arrived at our final category of algebraic structure, the field. The history of finite fields can be traced back to the 18th century.(Dong 2010) In modern research finite fields have assumed a role of fundamental importance because of their practical applications in a wide variety of areas such as "coding theory, cryptography, algebraic geometry and number theory".(Dong 2010 ch4s4) With this structure, we may now begin to consider the mechanism of AES byte arithmetic, or application of the arithmetic operations of addition, negation, multiplication and inversion, on bytes. To model the relationship between the arithmetic operations of addition and multiplication as well as their respective inverses, via a single algebraic structure we examine a set and two operations by which we can establish an additive and a multiplicative group.(Paar & Pelzi 2009) This is what abstract algebraic theory defines as a field. A field consists of a set F, with two invertible binary operations defined on its elements.(Jackson 2017) A given field F, and it's operations which we will denote '+' and '×' are represented by the triple (F, +, ×).(Jackson 2017) In order to qualify as a field, an underlying set and it's operations are required to satisfy the following properties:

Field

(F, +, ×) is a field iff (F, +) and (F\{0}, ×) are abelian groups, such that . < F, +, · > is a commutative ring and × is distributive over +. (Paar & Pelzi 2009)

+,×	Associative:	$\forall a,b,c \in F\ (a+b)+c = a+(b+c)$ and $(a\times b)\times c = a\times(b\times c)$
+,×	Commutative:	$\forall a,b \in F\ a+b = b+a$ and $a\times b = b\times a$
+,×	Identity:	$\forall a \in F\ \exists 0,1 \in F\ a+0 = a$ and $a\times 1 = a$
+	Inverse:	$\forall a \in F\ \exists \text{-}a \in F\ such\ that\ a+(\text{-}a)=0$
×	Inverse:	$\forall a \in F \setminus \{0\}\ \exists a^{-1} \in F\ such\ that\ a\times a^{-1}=1$
+,×	Distributive:	$\forall a,b,c \in F\ (a+b)\times c = a\times c + b\times c$

When we exclude the additive identity, the field elements under addition + and under multiplication × both form commutative groups. As explained previously, a field is a commutative ring in which there exists an inverse ∀a ∈ F\{0}). ∀a∈F\{0} means for all elements a of F, excluding 0. We exclude 0, the additive identity, as this element has no inverse under × by definition. We have shown how sets of nonprime order with multiplication modulo n do not define a group, for example, $\mathbb{Z}_8$/{0} = {1, 2, 3, 4, 5, 6, 7}. Therefore, ($\mathbb{Z}_n$, +, *), in general, would

not represent a field. However, for prime n, $n \in \mathbb{P}$, the structure formed $Z^*_{p,}$ is an Abelian group. For example $Z_5/\{0\} = \{1, 2, 3, 4\}$ $\mathbb{Z}_5^*$. $(Z^*_p, +, *)$ defines a unique field type, known as a prime field. These are fields with a prime, non-infinite order, denoted $\mathbb{Z}/p\mathbb{Z}$, or F_p.[Paar & Pelzi 2009] This means that if we consider the set $\mathbb{Z}_n$ along with modular addition and multiplication, for prime n $n \in \mathbb{P}$, $\mathbb{Z}_n$ is not only a commutative ring but also a prime field.[Paar & Pelzi 2009] Let p be a prime, $\forall p \in \mathbb{P}$, the field of order p, may be constructed as the integers modulo p with members represented by $\mathbb{Z}_p = \{0, 1, ..., (p-1)\}$.[Paar & Pelzi 2009] Thus, a prime field is the field of equivalence classes modulo p, $a = b$ in F_p means the same as $a \equiv b \pmod p$.

6.7 : Finite Fields

Cryptographic algorithms, both symmetric and asymmetric ciphers, are typically concerned with arithmetic on a finite number of elements, thus fields with a finite number of elements are of interest.(Paar & Pelzi 2009) A finite field consists of a finite set F, that defines arithmetic for two invertible binary operations.(Jackson 2017) A given finite field GF, of order n, and its operations, which we will denote '+' and '×', are represented by the triple (GF(n), +, ×), where the letters GF stand for Galois Field. Finite fields are known as Galois Fields, in honour of Évariste Galois, a French mathematician who established finite fields and proved:(Jackson 2017)

Finite Field

A field of order q exists iff $q = p^k$ $\forall p \in \mathbb{P}$, $k \in \mathbb{Z}+$

Prime fields are of the form p^k where the field characteristic $p \in \mathbb{P}$ and k = 1, however, these are not the only fields of finite membership. An integer q produced by exponentiation of a prime p^k where $p \in \mathbb{P}$ and k = 1, is known as a prime power. Therefore Finite or Galois fields are those fields with a finite order equal to a prime or prime power.

We now see that there exists finite fields with 5 (5^1) members and 343 (7^3) members. In fact, the field used by the AES has 256 members which can be seen to be a prime power, 256 = 2*2*2*2*2*2*2*2 = 2^8. However, there is no finite field with order 12 as 12 = 2*2*3, and 12 is thus not a prime power.

All finite fields of the same order are structurally identical, $\forall p \in \mathbb{P}$, $\forall n \in \mathbb{N}$ finite fields $GF(p^n)$ of equivalent order are unique up to isomorphism.(Jackson 2017) Isomorphic structures display equivalent algebraic behavior.(Daemen & Rijmen 2002)

Finite fields $GF(p^n)$ $\forall p \in \mathbb{P}$, $\forall n \in \mathbb{N}$, where n>1, can be represented in several ways.(Daemen & Rijmen 2002) The representation of $GF(p^n)$ members by means of polynomials with coefficients over F_p is the method implemented by the specification of the AES.(Daemen & Rijmen 2002) As well, for finite fields with an order that are a prime power, the additive and multiplicative operations cannot be executed as modular arithmetic.(Daemen & Rijmen 2002) In the next sections, we explain this representation and it's a form of arithmetic.

Section 7 : Galois Field Arithmetic

7.1 : Prime Fields

The transformations implemented by the AES operate within the Galois Field of 256 members known as $GF(2^8)$.(Paar & Pelzi 2009) In more detail, Rijndael's Galois Field, which the AES implements, only allows numbers representable by 8 bits. Due to the property of closure, we are assured that all mathematical operations defined in the field result in an 8-bit number. Thus operands and resultants are limited to the range $0 \leq i < 2^8$, represented by numbers from 0 to 255

Within this algebraic structure we redefine the arithmetic operations of addition, subtraction, multiplication, and division such that these operations, when performed on the underlying set, remain consistent with the behaviour of an infinite set.(Paar & Pelzi 2009) When a system such as this has been properly defined, it allows us to perform finite field arithmetic, that is, perform operations adhering to the necessary arithmetic laws with a field of finite members as opposed to infinite, like the fields of rational or real numbers.(Paar & Pelzi 2009) Galois Field Arithmetic has become integral to many modern applications such as linear block codes in coding theory, Reed–Solomon error correction, and in cryptography algorithms such as the AES encryption algorithm.(Gallian 2016) While, by definition, no finite field is infinite, there are infinitely many finite fields. This section describes how these arithmetic operations have been defined for the AES.

As we have shown, the order of a finite field is necessarily of the form p^n for $p \in \mathbb{P}$, $n \in \mathbb{Z}^+$. Our first example of finite field arithmetic, and perhaps the most intuitive, are operations within fields of prime order p^n, where $p \in \mathbb{P}$ and n = 1 e.g. $GF(p)$ = {0, 1, . . . , (p − 1)}. Operations in a prime field $GF(p)$ employ the system of modular arithmetic, using the field characteristic as the modulus.(Daemen & Rijmen 2002) Thus the prime field operations are "integer addition modulo p"(Daemen & Rijmen 2002 p. 13) and "integer multiplication modulo p".(Daemen & Rijmen 2002 p. 13) We consider the finite field of order 5, GF(5) = {0,1,2,3,4}. The additive inverse is $a + (-a) = 0 \bmod p$, while any nonzero multiplicative inverse is given by $a \cdot a^{-1} = 1$. Tables 13, 14, 15, and 16, shown below, describe binary addition and multiplication, as well as the additive and multiplicative inverse of each field element.

addition

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

additive inverse

-0 = 0
-1 = 4
-2 = 3
-3 = 2
-4 = 1

multiplication

*	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

multiplicative inverse

0^{-1} DNE
$1^{-1}=1$
$2^{-1}=3$
$3^{-1}=2$
$4^{-1}=4$

Tables 13, 14, 15, 16

As all finite fields used in the description of the AES have a characteristic of 2, the mechanisms of arithmetic in $GF(2)$ are necessary to understand. Using the characteristic of $GF(2)$ for modular arithmetic, the only possible values are 0 and 1, as a modulus of 2 only permits these integer remainders. Thus $GF(2)$ (also F_2, $\mathbb{Z}/_2\mathbb{Z}$ or $\mathbb{Z}_2$) is the Galois Field of two members, $GF(2) = \{0, 1\}$, which happens to be the field of least order. The two elements are nearly always called 0 and 1, being the additive and multiplicative identities, respectively. As defined, arithmetic is done modulo 2, yielding Tables 17 and 18:

$GF(2)$ addition and subtraction correspond to the logical XOR operation.(Daemen & Rijmen 2002)

+/-	0	1
0	0	1
1	1	0

Table 17

$GF(2)$ multiplication corresponds to the logical AND operation.(Daemen & Rijmen 2002)

*	0	1
0	0	0
1	0	1

Table 18

7.2 : Extension Fields $GF(p^n)$

However, the field used by the AES has far more than two members. The AES implements the finite field containing 256 members, denoted GF(2^8) chosen specifically because each of the field elements can be represented by a single 8-bit byte.(Paar & Pelzi 2009) AES considers every operand byte as a GF(2^8) field member and all data manipulation performed by the AES is within this finite field.(Paar & Pelzi 2009)

In Section 6.2 we show that ($\mathbb{Z}$, +, *) does not form a field because ($\mathbb{Z}\backslash\{0\}$, *) is not a multiplicative group. In general, ($\mathbb{Z}_n$, +, *) is not a finite field because only those members coprime to the order have inverses. For example, $\mathbb{Z}_{10}/\{0\}$ = {1, 2, 3, 4, 5, 6, 7, 8, 9} under modular multiplication is not a group. When n is a prime number, $n \in \mathbb{P}$, addition and multiplication modulo a prime number p form a finite field.(Daemen & Rijmen 2002) $\mathbb{Z}_5/\{0\}$ ={1, 2, 3, 4} with modular multiplication is the Abelian group $\mathbb{Z}_5^*$, therefore, ($\mathbb{Z}_5$, +, *) is a finite field.(Daemen & Rijmen 2002) The characteristic of prime fields is the order, we shall show that modular arithmetic does not allow us to construct a finite field with order of p^n for $p \in \mathbb{P}$ and $n \in \mathbb{N}$ where n>1.

Fields, with an order that is a prime power p^n where n > 1, are known as Extension Fields.(Paar & Pelzi 2009) We will represent order p^n finite fields using $GF(p^n)$.(Daemen & Rijmen 2002) If the order of $GF(p^n)$ is not prime, the field operations cannot be represented by modular addition and multiplication.(Daemen & Rijmen 2002) We will see that extension fields members require both a notation for its members and a method of arithmetic. These members are represented as polynomials, and that computation in an extension field is achieved by performing modular polynomial arithmetic.

7.3 : Representation $GF(p^n)$

We represent an extension field using the polynomial basis.(Cid Murphy Robshaw 2006) By this method, a field's elements are representable by a set $GF(p^n)$ of order p^n and with two polynomial operations.(Paar & Pelzi 2009) The AES considers these polynomials as abstract entities only, while they are represented by polynomial equations, these equations are never evaluated.(Daemen & Rijmen 2002) As coefficient values are defined by a field F, we say it is a polynomial over F.(Cid Murphy Robshaw 2006) Elements of $GF(p^n)$ are thus represented as polynomials of degree less than n over GF(p). Generally, a polynomial over a field F is denoted:

$$b(x) = b_{n-1}x^{n-1} + b_{n-2}x^{n-2} + ... + b_2x^2 + b_1x + b_0$$

where x is known as the indeterminate of the polynomial, and the coefficients are members $b_i \in GF(2)$.(Daemen & Rijmen 2002) The degree of a polynomial is L if $b_i = 0, \forall i > L$, and L is the smallest number with this property.(Daemen & Rijmen 2002) We will denote the set of polynomials over a given field F by $F[x]$, the set of polynomials over a field F of degree below L is denoted by $F[x]|L$, such that the AES field GF(2^8) = GF(2)|8 .(Daemen & Rijmen 2002) "In computer memory, the polynomials in $F[x]|L$ with F a finite field can be stored efficiently by storing the L coefficients as a string".(Daemen & Rijmen 2002 p. 13)

7.4 : Rules of Arithmetic $GF(p^n)$

Now that we have a form of representation for extension field elements, we are able to define polynomial arithmetic. Polynomials over field $GF(p)$, are capable of addition and multiplication but the coefficients are computed in $GF(p)$ and must be reduced modulo p.[Paar & Pelzi 2009] For example, compare the result of the equations from Appendix A Polynomial Arithmetic under GF(11):[Dong 2010]

$$(x^5 + 3x^3 + 4)+(6x^6 + 4x^3) = 6x^6 + x^5 + 7x^3 + 4$$

$$(x^5 + 3x^3 + 4)-(6x^6 + 4x^3) = 5x^6 + x^5 +10x^3 + 4$$

$$(x^5 + 3x^3 + 4)*(6x^6 + 4x^3) = 6x^{11} + 7x^9 + 4x^8 + 3x^6 + 5x^3$$

$$(3x^6 + 7x^4 + 4x^3 + 5) \div (x^4 + 3x^3 + 4) = 3x^2 + 3x + 3 \text{ with remainder } x^3 + 10x^2 + 4x +1$$

If $a(x)= a_nx^n + a_{n-1}x^{n-1} + ... + a_0$ and $b(x)= b_mx^m + b_{m-1}x^{m-1} + ... + b_0$ are two polynomials over a field F, then $a(x) = p(x)*b(x)+r(x)$ for unique $r(x)$ $p(x) \in F$ where $r(x)$ is of degree smaller than n.[Dong 2010] The polynomial $r(x)$ is called the remainder of $a(x)$ modulo $b(x)$.[Dong 2010] For polynomials $a(x)$, $b(x)$, $p(x) \in F$, if $p(x)$ divides $a(x)-b(x)$, then $a(x)$ is congruent to $b(x)$ modulo $b(x)$.[Dong 2010] We denote the modular equivalence of polynomials in a similar fashion to the integers written $a(x) \equiv b(x) \bmod p(x)$.[Dong 2010]

A polynomial d(x) is irreducible over the field $GF(p)$ iff there exist no two polynomials a(x) and b(x) with coefficients in $GF(p)$ such that $d(x) = a(x) \times b(x)$, where a(x) and b(x) are of degree > 0.[Daemen & Rijmen 2002 p. 15] Irreducible polynomials are roughly comparable to prime numbers, i.e., their only factors are 1 and the polynomial itself.[Cid Murphy Robshaw 2006]

In contrast to prime fields, operation performed in an extension field, those fields of the form $GF(p^n)$, requires two moduli. These moduli are the polynomial modulus and the integer modulus. Arithmetic results must first be reduced by the polynomial modulus and then the remainder's coefficients must be reduced by the integer modulus. An example shown in Figure 16, below, is $2x^2+x^4 \equiv 0 \bmod (x^2-1)$ in GF(3)|2 is confirmed if we first reduce by the irreducible polynomial modulus x^2-1 to obtain a remainder of 3. This remainder is then reduced by the field characteristic 3: $3 \equiv 0 \bmod 3$ confirming the examples equivalence.[Dong 2010]

$$
\begin{array}{r}
x^2+3 \\
x^2-1\overline{)\ x^4+2x^2} \\
\underline{x^4-x^2} \\
3x^2 \\
\underline{3x^2-3} \\
3
\end{array}
$$

Figure 16

Extension field addition is executed similar to that of prime fields. In $GF(p)$, we perform addition of each polynomial's like terms, each term is reduced by the prime characteristic and our result is the remainder.(Paar & Pelzi 2009) Generally, under $GF(p^n)$, the same polynomial terms are summed then reduced using the prime field characteristic.(Paar & Pelzi 2009) The degree of c(x) is at most the maximum of the degrees of a(x) or b(x) and as the polynomial modulus used to construct the field limits their degree, the addition is not only closed but is performed without the need for reduction by the polynomial modulus.(Daemen & Rijmen 2002) Addition of polynomials is associative and commutative.(Daemen & Rijmen 2002) The neutral element or additive identity, 0, is the polynomial with all coefficients equal to 0.(Daemen & Rijmen 2002) The inverse of the polynomial basis is found by replacing each coefficient by its inverse element in F.(Daemen & Rijmen 2002) Thus the structure $< GF(p)|n,\ + >$ is an Abelian group.(Daemen & Rijmen 2002)

Extension field multiplication is executed via an approach similar to the case of multiplication in prime fields. In $GF(p)$, we multiply the two representative polynomials then reduce each term by the field characteristic and consider only the remainder.(Paar & Pelzi 2009) In extension fields, the product of the multiplication is divided by an irreducible polynomial, and we consider only the remainder after reduction by the field characteristic.(Paar & Pelzi 2009) With respect to polynomial addition, polynomial multiplication is associative, commutative, and distributive.(Daemen & Rijmen 2002) In order to make the multiplication closed over $GF(p)|n$, we select a polynomial m(x) of degree n.(Daemen & Rijmen 2002) The multiplication of two polynomials a(x) and b(x) is then defined as the algebraic product of the polynomials modulo m(x): $c(x) \equiv a(x) \times b(x)\ (mod\ m(x))$.(Daemen & Rijmen 2002) The group identity or neutral element is the polynomial of degree 0 and with coefficient of x^0 equal to 1.(Daemen & Rijmen 2002) Distributive As well, it holds that $a(x) \bullet (b(x) + c(x)) = a(x) \bullet b(x) + a(x) \bullet c(x)$.(Daemen & Rijmen 2002) The structure $< GF(p)|n,\ +,\ \cdot\ >$ is a commutative ring.(Daemen & Rijmen 2002)

We now show that for special choices of the reduction polynomial m(x), the $GF(p)|n$ structure is a field.(Daemen & Rijmen 2002) Similar to standard modular arithmetic, the multiplicative inverse is found by means of the Extended Euclidean Algorithm (EEA).(Daemen & Rijmen 2002) Given a polynomial

$a(x)$ for which we would like the inverse, the EEA provides polynomials $b(x)$ and $c(x)$ such that $a(x)\times b(x) + m(x)\times c(x) = gcd(a(x), m(x))$.(Daemen & Rijmen 2002) $gcd(a(x), m(x))$, the greatest common divisor of the polynomials $a(x)$ and $m(x)$, is always equal to 1 iff $m(x)$ is irreducible.(Daemen & Rijmen 2002) Applying modular reduction with an irreducible $m(x)$ we get: $a(x) \times b(x) \equiv 1 \ (mod\ m(x))$.(Daemen & Rijmen 2002) By algebraic manipulation we find that $b^{-1}(x) = a(x\ )\ mod\ m(x)$ and thus b(x) is the inverse element of a(x).(Daemen & Rijmen 2002) Therefore, if we let F be the field $GF(p^n)$, with an irreducible modulus m(x) of degree n over $GF(p)$, the structure $< GF(p)|n,\ +,\ \cdot\ >$ is a field of order p^n, represented by the polynomial basis over $GF(p)$ of degree less than n.(Dong 2010)(Daemen & Rijmen 2002) For example, we can represent the extension field $GF(3^2)$ by a polynomial basis of degree less than 2 using the irreducible polynomial $x^2 + 1$, shown by Table 19 and 20 below.(Dong 2010)

+	0	1	2	x	x+1	x+2	2x	2x+1	2x+2
0	0	1	2	x	x+1	x+2	2x	2x+1	2x+2
1	1	2	0	x+1	x+2	x	2x+1	2x+2	2x
2	2	0	1	x+2	x	x+1	2x+2	2x	2x+1
x	x	x+1	x+2	2x	2x+1	2x+2	0	1	2
x+1	x+1	x+2	x	2x+1	2x+2	2x	1	2	0
x+2	x+2	x	x+1	2x+2	2x	2x+1	2	0	1
2x	2x	2x+1	2x+2	0	1	2	x	x+1	x+2
2x+1	2x+1	2x+2	2x	1	2	0	x	x+2	x+2
2x+2	2x+2	2x	2x+1	2	0	1	x+2	x	x+1

Table 19

+	0	1	2	x	x+1	x+2	2x	2x+1	2x+2
0	0	0	0	0	0	0	0	0	0
1	0	1	2	x	x+1	x+2	2x	2x+1	2x+2
2	0	2	1	2x	2x+2	2x+1	x	x+2	x+1
x	0	x	2x	2	x+2	2x+2	1	x+1	2x+1
x+1	0	x+1	2x+2	x+2	2x	1	2x+1	2	x
x+2	0	x+2	2x+1	2x+2	1	x	x+1	2x	2
2x	0	2x	x	1	2x+1	x+1	2	2x+2	x+2
2x+1	0	2x+1	x+2	x+1	2	2x	2x+2	x	1
2x+2	0	2x+2	x+1	2x+1	x	2	x+2	1	2x

Table 20

7.5 : Representation GF(2^n)

For any prime power there is a single finite field, hence all representations of $GF(p^n)$ are isomorphic.(Paar & Pelzi 2009) Despite this equivalence, the representation has an impact on the implementation complexity.(Paar & Pelzi 2009) Thus the AES finite field members can be represented in several different ways. When the prime characteristic is 2, it is conventional to express elements of $GF(p^n)$ as binary numbers, with each term in a polynomial represented by one bit in the corresponding element's binary expression.(Daemen & Rijmen 2002)

Extension fields are called binary fields or characteristic-two fields if their order is of the form 2^n.(Dong 2010) Fields with a characteristic of two are particularly efficient for implementation in modern day, binary based, hardware and software.(Dong 2010) This is because any extension field with a characteristic of two, GF(2^n), has elements that, rather than being represented as integers, are representable as polynomials with coefficients in GF(2) = {0, 1}, known as binary polynomials.(Dong 2010) GF(2^n) elements can be represented as n-bit strings with each bit position corresponding to polynomial coefficients of the same position.(Dong 2010) Members of this field represented by the polynomial basis have a maximum degree of (n−1), so that there are n coefficients in total for every element and 2^n such polynomials in the field.(Dong 2010) For the field $GF(2^8)$, which is implemented by the AES, there are exactly 2^8 = 256 such polynomials.(Paar & Pelzi 2009) Each polynomial representation is a bitstring consisting of the bits $\{b_7\ b_6\ b_5\ b_4\ b_3\ b_2\ b_1\ b_0\}$ represented by a polynomial basis over $GF(2) = \{0, 1\}$:

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = \sum_{1=0}^{7} b_ix^i$$

Example: the hexadecimal value ‘57’ (binary 01010111) corresponds with polynomial (NIST 2001)

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 \rightarrow$$
$$0 * x^7 + 1 * x^6 + 0 * x^5 + 1 * x^4 + 0 * x^3 + 1 * x^2 + 1 * x + 1 \rightarrow x^6 + x^4 + x^2 + x + 1$$

The following are equivalent representations in a characteristic 2 finite field:(NIST 2001)

Polynomial: $x^6 + x^4 + x^2 + x + 1$ Binary: {01010111} Hexadecimal: {57}

During the operation of the AES, the most basic data objects we manipulate are 8-bit bytes.[(NIST 2001)] All possible byte values may be expressed by the polynomial basis of GF(2) with degree eight or fewer.[(NIST 2001)] Due to all members of GF(2^8) being representable by a single byte length bitstring representing the polynomial basis over GF(2), AES byte arithmetic is represented by polynomial arithmetic over the field GF(2).[(NIST 2001)] Addition and Multiplication of bytes is executed by polynomial arithmetic of corresponding polynomial coefficients in the underlying field of GF(2) modulo an irreducible binary polynomial of degree 8.[(Paar & Pelzi 2009)] Therefore, to define byte arithmetic, the AES provides a suitable reduction polynomial m(x) given by:[(Paar & Pelzi 2009)]

$$m(x) = (x^8 + x^4 + x^3 + x + 1)$$ or 11B in hexadecimal notation

7.6 : Addition GF(2^8)

The AES key schedule and key addition layer rely on addition.(Paar & Pelzi 2009) The AES implements addition in the extension field GF(2^8).(Daemen & Rijmen 2002) Extension field addition is generally executed by first using a polynomial basis to represent field elements, then performing modular arithmetic in the underlying field.(Paar & Pelzi 2009) For the AES, polynomial coefficients are handled in the underlying field GF(2).(Paar & Pelzi 2009) In GF(2), the sum of two members is given by their integer sum modulo 2.(Paar & Pelzi 2009) We have seen that addition and subtraction modulo 2 are the same operation as displayed in Table 17. GF(2) is defined by modulo 2 arithmetic by which addition and subtraction are defined to be the same operation as bitwise XOR. Therefore the operation of polynomial summation and its inverse consists of an XOR between the coefficients of equal-powered terms. Let A(x),B(x) ∈ GF(2^n), we represent their sum or difference by:(Paar & Pelzi 2009)

$$C(x) = A(x) + B(x) = \sum_{i=0}^{n-1} c_i x^i \equiv a_i + b_i \ mod\ 2$$

$$C(x) = A(x) - B(x) = \sum_{i=0}^{n-1} c_i x^i \equiv a_i - b_i \ mod\ 2$$

An example in GF(2^8) is the sum of the polynomials {57} and {83} computed:(Paar & Pelzi 2009)

$$A(x) = x^6 + x^4 + x^2 + x + 1$$
$$B(x) = x^7 + x + 1$$
$$C(x) = x^7 + x^6 + x^4 + x^2$$

7.7 : Multiplication in GF(2^8)

Extension field multiplication is the core operation of the AES MixColumn function.(Paar & Pelzi 2009) The AES implements multiplication in the field GF(2^8).(Paar & Pelzi 2009) Extension field multiplication is generally executed by first using a polynomial basis to represent field elements, then performing modular arithmetic in the underlying field.(Paar & Pelzi 2009) For the AES, polynomial coefficients are handled in the underlying field GF(2).(Paar & Pelzi 2009) We use the "•" symbol to denote extension field multiplication. GF(2^n) extension field elements represented by a GF(2) polynomial basis are multiplied using arithmetic polynomial multiplication:(Paar & Pelzi 2009)

$$C(x) = A(x) \cdot B(x) \; = \; (a_{m-1}x^{m-1} + ... + \; a_0) \cdot (b_{m-1}x^{m-1} + ... + \; b_0) \; = \; (c_{2m-2}x^{2m-2} + ... + \; c_0)$$

where: $c_0 = a_0 b_0 \; mod \; 2$, $\quad c_1 = a_0 b_1 + a_1 b_0 \; mod \; 2$, ... , $\quad c_{2m-2} = a_{m-1} b_{m-1} \; mod \; 2$

The coefficients a_i, b_i and $c_i \in$ GF(2), and that coefficient arithmetic is performed in GF(2).(Paar & Pelzi 2009) A product polynomial C(x) with degree higher than (n−1) must be reduced such that results are representable by the AES polynomial basis of degree 7 and below. Thus the AES supplies an irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$ of degree n with coefficients from GF(2). An example in GF(2^8) is the product of the elements denoted by {57} and {83}:(Gladman 2003)

{57} • {83} = {C1}, or:

$$(x^6 + x^4 + x^2 + x + 1) \; + \; (x^7 + x + 1)$$
$$= (x^{13} + x^{11} + x^9 + x^8 + x^7) \oplus (x^7 + x^5 + x^3 + x^2 + x) \oplus (x^6 + x^4 + x^2 + x + 1)$$
$$= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$$

$$x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \; modulo \; x^8 + x^4 + x^3 + x + 1 = x^7 + x^6 + 1$$

7.8 : Polynomials with Coefficients in GF(2^8)

In the AES specification, data words consisting of 4-byte columns of the state array can be modeled by polynomials over GF(2^8) of the form: $a(x) = a_3x^3 + a_2x^2 + a_1x + a_0$.[NIST 2001] These polynomials have a degree of four with coefficients that are values of the AES extension field. We will denote such a collection by a 4-byte vector [a_0, a_1, a_2, a_3].[NIST 2001] These polynomials behave differently than polynomials previously used to represent individual field elements. While both polynomial definitions use the same indeterminate, x, the coefficients of this new definition are extension field elements defined by bytes, instead of bits.[NIST 2001] The multiplication of four-term polynomials uses a different reduction polynomial, defined below.[NIST 2001]

Addition of these new word value representations is performed by adding the extension field coefficients of like powers of the indeterminate x.[NIST 2001] If we let a(x) and b(x) be polynomials over $GF(2^8)$:

$$a(x) = a_3x^3 + a_2x^2 + a_1x + a_0 \text{ and } b(x) = b_3x^3 + b_2x^2 + b_1x + b_0$$

then the addition operation is a combination of corresponding bytes of each word, the XOR of the complete word values:

$$a(x) + b(x) = (a_3 \oplus b_3)x^3 + (a_2 \oplus b_2)x^2 + (a_1 \oplus b_1)x^1 + (a_0 \oplus b_0)x^0$$

Multiplication of these new word value representations is performed in two steps.[NIST 2001] If we let a(x) and b(x) be polynomials over $GF(2^8)$:

$$a(x) = a_3x^3 + a_2x^2 + a_1x + a_0 \text{ and } b(x) = b_3x^3 + b_2x^2 + b_1x + b_0$$

then, first, their polynomial product $c(x) = a(x)b(x)$ is algebraically expanded.[NIST 2001] coefficients of same powered terms are collected to give:

$$c(x) = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x^1 + c_0$$

$$c_0 = a_0 \bullet b_0 \qquad c_1 = a_1 \bullet b_0 \oplus a_0 \bullet b_1 \qquad c_2 = a_2 \bullet b_0 \oplus a_1 \bullet b_1 \oplus a_0 \bullet b_2$$
$$c_3 = a_3 \bullet b_0 \oplus a_2 \bullet b_1 \oplus a_1 \bullet b_2 \oplus a_0 \bullet b_3 \qquad c_4 = a_3 \bullet b_1 \oplus a_2 \bullet b_2 \oplus a_1 \bullet b_3 \qquad c_5 = a_3 \bullet b_2 \oplus a_2 \bullet b_3$$
$$c_6 = a_3 \bullet b_3$$

Often, $c(x)$ can no longer be represented by a 4-byte vector so we must reduce $c(x)$ modulo a polynomial of degree 4.(NIST 2001) For the AES algorithm, this is accomplished with the polynomial $x^4 + 1$, so that $x^i \bmod (x^4 + 1) = x^{(i \bmod 4)}$.(NIST 2001) The modular product of a(x) and b(x), denoted $a(x) \otimes b(x)$, is given by the four-term polynomial d(x):(NIST 2001)

$$d(x) = d_3 x^3 + d_2 x^2 + d_1 x + d_0$$

$$d_0 = (a_0 \bullet b_0) \oplus (a_3 \bullet b_1) \oplus (a_2 \bullet b_2) \oplus (a_1 \bullet b_3)$$
$$d_1 = (a_1 \bullet b_0) \oplus (a_0 \bullet b_1) \oplus (a_3 \bullet b_2) \oplus (a_2 \bullet b_3)$$
$$d_2 = (a_2 \bullet b_0) \oplus (a_1 \bullet b_1) \oplus (a_0 \bullet b_2) \oplus (a_3 \bullet b_3)$$
$$d_3 = (a_3 \bullet b_0) \oplus (a_2 \bullet b_1) \oplus (a_1 \bullet b_2) \oplus (a_0 \bullet b_3)$$

Figure 17

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Figure 18

AES operations consisting of multiplication by a fixed polynomial $a(x)$ may be written as the matrix multiplication, shown in Figure 18, above and solved for by the system of equations shown in Figure 17, above. Because $x^4 + 1$ is not an irreducible polynomial over $GF(2^8)$, $x^4 + 1 = (x + 1)^4$ multiplication by a fixed polynomial is not necessarily invertible.(NIST 2001) A polynomial $a(x)$ has an inverse if the polynomial $x + 1$ does not divide it.(NIST 2001) For these operations, the AES algorithm specifies a fixed four-term polynomial that does have an inverse:(NIST 2001)

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} \qquad a^{-1}(x) = \{0B\}x^3 + \{0D\}x^2 + \{09\}x + \{0E\}$$

Part 3

A Specification of the Advanced Encryption Standard

Section 8 : Notation and Conventions

8.1 : Inputs and Outputs

The specified input parameters and output resultants of the AES algorithm are represented by sequences of binary digits, or "bit" values.(NIST 2001) Binary values are sequences defined by a base-2 numeral system represented by the concatenation of any mechanism possessing one of two exclusive states as described in Section 1.1. Bit sequences in this document will be represented as ordered series comprised of 1 and 0 states.

When we act upon a fixed length series of bits we speak of processing a singular "block".(NIST 2001) The number of bits a given fixed length series contains is then known as it's block length, or block size.(Paar & Pelzi 2009) The block length specified by the AES is 128 bits, meaning that the plaintext and ciphertext blocks processed and output by the AES may only be 128 bits in size.(NIST 2001) Contrarily, the cipher key length specified by the AES may vary between 128, 192 or 256 bits but must remain fixed during a given execution.(NIST 2001)

The AES algorithm is a subset of the Rijndael block cipher.(Daemen & Rijmen 2002) Rijndael supports variable block and key lengths, allowing both to be specified as any multiple of 32, between 128 and 256 bits.(Daemen & Rijmen 2002) The reason for AES's specific parameterization of the Rijndael cipher is due to the focus of testing during the AES selection process.(NIST 2001) As other configurations were not subject to the same rigorous testing and peer review, they are not permitted by this standard.(LFS 2018)

The location of each bit in a given sequence is defined using a zero-based ordering.(NIST 2001) The initial element is assigned a base index of 0 and each subsequent element is indexed by the subsequent natural number from 0 to $(sequence\ length\ -\ 1)$. This distinction is represented by subscripting the index, i. Thusly, a single bit is specified by b_i, where $b \in \{1, 0\}$ and $0 \leq i < 128$, $0 \leq i < 192$ or $0 \leq i < 256$ depending on the configuration(NIST 2001)

The smallest bit sequence, or basic processing unit, addressed by the operations comprising the AES is known as a byte.(NIST 2001) A byte most commonly consists of a length eight bit sequence representative of a single entity, in this case we implement eight bit bytes as operands and resultants during the computations performed by AES.

AES byte value representations result from concatenation of eight bits enclosed by braces:

$$\{ b_7,\ b_6,\ b_5,\ b_4,\ b_3,\ b_2,\ b_1,\ b_0 \}$$

where individual bit values $b \in \{0,\ 1\}$. During the computations performed by AES, bytes values are interpreted as finite field elements of polynomial representation:

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = \sum_{i=0}^{7} b_ix^i$$

For example, $\{10100011\}$ identifies the specific finite field element $x^7 + x^5 + x + 1$.

Binary-coded values of increasing length become difficult for human comprehension and manipulation. Hexadecimal numerals of base 16 are a popular alternative, allowing direct conversion from four binary digits to a single hexadecimal digit. Table 21 provides a reference for this conversion. Hexadecimal representation of byte values is convenient. A series of eight bits is divided into the four leftmost higher indexed bits and the four rightmost lower indexed digits. A single byte value in the range of [0000 0000, 1111 1111] then becomes [00, ff] under hexadecimal representation.

For example, the byte value $\{01010011\}$ is denoted $\{A3\}$.

Bit Pattern	Character
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Table 21

8.2 : Arrays

During the execution of AES, byte operands are held in abstract collections known as arrays.(NIST 2001) Formally, arrays are zero indexed data structures representing disk information with some mechanism of addressability by the CPU, necessary for data processing.(Greenlaw & Hoover 1998) While an array may hold any data object, byte arrays partition bit sequences into contiguous groups of eight, providing each with an index n. The block and key lengths described previously, specifically 128, 192, and 256 bits, may also be referenced by their byte lengths 16, 24, and 32 respectively.(NIST 2001)

For AES operand and resultant arrays, denoted by a, constituent bytes are referenced in two ways. Similar to the representation of individual elements of a bit sequence, the nth-element of a given array a is expressed a_n, where n is the array index, as well, $a[n]$ also expresses an individual byte of index n. For either representation the range of n reflects cipher block or key length: 128 bits, $0 \leq n < 16$; 192 bits, $0 \leq n < 24$; 256 bits, $0 \leq n < 32$.(NIST 2001)

Arrays of bytes are represented:(NIST 2001 p. 8)

$$a_0\ a_1\ a_2\ \ldots\ a_{15}$$

The ordering of individual bits can be seen from the 128-bit input sequence:(NIST 2001 p. 8)

$$input_0 input_1 input_2\ \ldots\ input_{126} input_{127}$$

An array of bytes and the bit locations within:(NIST 2001 p. 9)

$$\begin{aligned} a_0 &= \{input_0,\ input_1,\ \ldots,\ input_7\}; \\ a_1 &= \{input_8,\ input_9,\ \ldots, input_{15}\}; \\ &\vdots \\ a_{15} &= \{input_{120},\ input_{121},\ \ldots,\ input_{127}\}; \end{aligned}$$

The pattern in general:(NIST 2001 p. 9)

$$a_n = \{input_{8n}, input_{8n+1}, \ldots, input_{8n+7}\} .$$

Input bit sequence	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	...
Byte Number	0								1								2								...
Byte Position	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	...

Table 22

8.3 : The State

Cipher transformations performed by the AES are modeled by a two dimensional array called the State. Two dimensional array position is specified by the use of two indices. The State consists of four rows of bytes, each containing Nb bytes, where Nb is the block length divided by 32.(NIST 2001) To express an element of the State s, we use $s_{r,\,c}$ or $s[r, c]$ where s is a given State, r is the row number $0 \le r < 4$ and c is the column number $0 \le c < Nb$.(NIST 2001) This standard specifies a single block length of 128 bits, thusly Nb is fixed at 4 and the value of c is always in the range $0 \le c < 4$.(NIST 2001)

Upon execution, the AES maps the byte elements of input blocks, $\{in_0,\ in_1,\ ...,\ in_{15}\}$ onto the state array in the order $s_{0,\,0},\ s_{1,\,0},\ s_{2,\,0},\ s_{3,\,0},\ s_{0,\,1},\ s_{1,\,1},\ s_{2,\,1},\ s_{3,\,1}...$, this transformation is modeled by:(NIST 2001)

$$s[r,\ c]\ =\ in[r\ +\ 4c] \qquad for\ 0 \le r < 4 \quad and \quad 0 \le c < Nb$$

AES transformations perform necessary encryption or decryption and the state bytes are mapped onto the output array, $\{out_0,\ out_1,\ ...,\ out_{15}\}$:(NIST 2001)

$$out[r\ +\ 4c]\ =\ s[r,\ c] \qquad for\ 0 \le r < 4 \quad and \quad 0 \le c < Nb$$

This process is depicted in Figure 19, below.

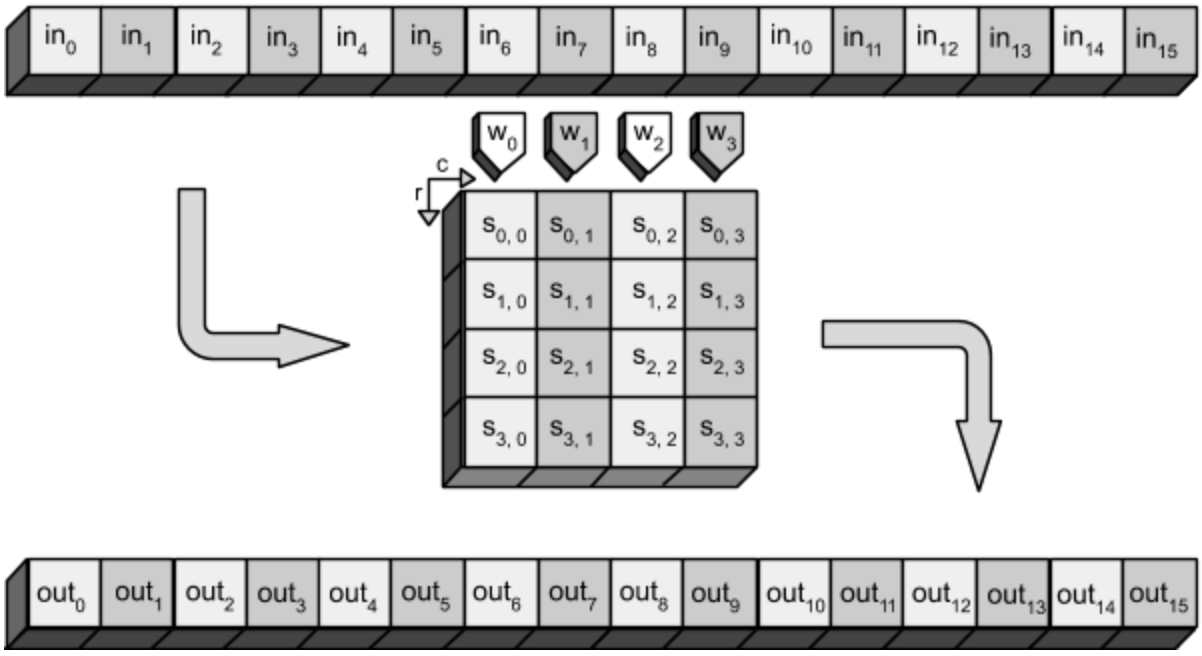

Figure 19

The State can be represented as an array of it's columns.(NIST 2001) While 8-bit bytes are the smallest operand addressed by AES, many of the procedures operate on an entire State column.(NIST 2001) As the AES fixes row number r at four, a State column possesses 4 bytes and is thus a 32-bit vector.(NIST 2001)

This specification defines 32-bit vectors as a single "word", and provides an alternate definition of the State as a one dimensional array of 32-bit words, of Nb length.(NIST 2001) The column number c is then the index of a given word array, while row number r indexes an individual word's constituent bytes.(NIST 2001) A single word is specified by w_i, where $0 \leq i < 4$ and the State may be defined as an array of four words $\{w_0, w_1, w_2, w_3\}$ with constituent bytes:

$$w_0 = \{s_{0,0}, s_{1,0}, s_{2,0}, s_{3,0}\}$$

$$w_1 = \{s_{0,1}, s_{1,1}, s_{2,1}, s_{3,1}\}$$

$$w_2 = \{s_{0,2}, s_{1,2}, s_{2,2}, s_{3,2}\}$$

$$w_3 = \{s_{0,3}, s_{1,3}, s_{2,3}, s_{3,3}\}$$

128-bit plaintext and ciphertext blocks can also be defined as consisting of $Nb = 4$ 32-bit words. Cipher keys are also defined by their word length, consisting of Nk 32-bit words, where

$Nk = \{4, 6, 8\}$. The AES has specified the key lengths of 128, 192, and 256 bits defining key spaces of $\mathbb{Z}_2^{128}$ $\mathbb{Z}_2^{192}$ $\mathbb{Z}_2^{256}$.(Daemen & Rijmen 2002) The Rijndael cipher, of which AES is a subset, has a key space of $\mathbb{Z}_2^{32*Nk}$.(Daemen & Rijmen 2002)

Like Rijndael the AES is a key-iterated block cipher, the state undergoes a number of transformation rounds, or iterations, defined by the key length. The number of these rounds is denoted Nr , where Nr is 10, 12, or 14, for Nk of 4, 6, or 8.(NIST 2001) See the effect of Nk in Figure 20 below.

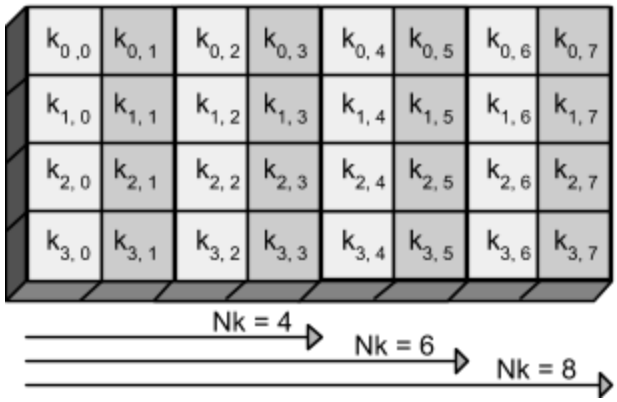

Figure 20

8.4 : The Substitution Box (S-box)

The Substitution box (S-box) is a cryptographic primitive, typically included as a component in block cipher design. This document will address the AES implementation alone. Desirable is a substitution which makes the relationship between output ciphertext and input key augmented plaintexts as complex as possible. This property, known as confusion, is one of two governing the operation of a secure cryptographic cipher first identified by Claude Shannon in his 1945 report A Mathematical Theory of Cryptography.(Shannon 1945)

In general, an n×m S-Box, input blocks of size n are substituted with output blocks of size m, implemented as a lookup table with 2^n m-bit members. We study the structure of the particular S-box implemented by the Rijndael cipher, of which the Advanced Encryption Standard (AES) cryptographic algorithm is a subset. This S-Box is based on an invertible transformation applied on the Galois field $GF(2^8)$. This transformation generates an 8×8 S-box, $S(x):\ GF(2^8) \rightarrow GF(2^8)$, implemented as a $2^8 = 256$ member square lookup table of size 16.The AES S-box models a function on polynomials over $GF(2)$ denoted $b' = S(b)$ mapping representative 8-bit inputs b, to 8-bit outputs b'.

Invented by Kaisa Nyberg, the AES S-box transformation is known as the "Nyberg S-box".(Nyberg 1991) The transformation can be expressed by the function $b' = Ab^{-1} \oplus c$.(NIST 2001) We find the multiplicative inverse of the Input in Rijndael's finite field, $GF(2^8) = GF(2)|[8]\ (x^8 + x^4 + x^3 + x + 1)$, then perform an affine transformation.(Daemen & Rijmen 2002) The S-Box transformation is be expressed in matrix form by Figure 21 below:

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad A= \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad c = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Figure 21

where bits $\{b_7, ..., b_0\}$ are the multiplicative inverse b^{-1}, and bits $\{b'_7, ..., b'_0\}$ are b', the result.

However, this equation is not to be interpreted as the sort matrix multiplication used for linear algebra computations. As product vector members would incorrectly be the summation of each row member multiplied by the column member of corresponding index, as described in Section 5.7. In the case of the $GF(2^8)$ representation selected for the AES, we recall that the addition of c is a bitwise addition $mod\ 2$, or bitwise XOR and multiplication is done modulo $(x^8 + x^4 + x^3 + x + 1)$, as described in Sections 7.7, 7.8. Product vector members are the XOR of each result of modular multiplication between row and column members of corresponding index. For example, $\{95\}$ has a multiplicative inverse in $GF(2^8)$ of $\{95\}^{-1} = \{8A\}$, or $\{10001010\}$ in binary.(NIST 2001) The full transformation of is shown by Figure 22 below:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Figure 22

The entity which models a cryptographic S-box is implemented as a table lookup, defined by a square matrix of size 16, shown by Table 23, below.(NIST 2001) The AES S-box lookup values can then be constructed in three steps by first supplying all potential input values over the Rijndael's Galois field ($GF(2^8)$) then composing two transformations:(NIST 2001)

1. Initialize the S-box with all values in $GF(2^8)$ represented by 256 byte values b_i = {0, 1, …, 255} in ascending sequence. When represented in a square matrix the first row contains {00}, {01}, …, {0F}; the second {10}, {11}, … {1F}; and so on. By this method, the value of the byte at column x, row y, is {xy}. These values then undergo two transformations.

2. We first map the value {00} to itself, then for each subsequent value we determine the multiplicative inverse in the finite field $GF(2^8)$.

3. Each output byte produced is considered as the 8 bit collection $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$, to be transformed by the following $GF(2)$ affine matrix:(NIST 2001)

$$b_i' = b_i \oplus b_{(i+4) mod 8} \oplus b_{(i+5) mod 8} \oplus b_{(i+6) mod 8} \oplus b_{(i+7) mod 8} \oplus c_i \qquad \text{for } 0 \le i < 8,$$

Where c_i is the ith bit of the value $\{63\}$ $\{c_7, c_6, c_5, c_4, c_3, c_2, c_1, c_0\} = \{01100011\}$ (NIST 2001)

To produce the inverse S-box seen in Table 24 we reverse the operations of the S-box, by first calculating the inverse affine transformation followed by the multiplicative inverse, as in Figure 23, below.(NIST 2001)

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Figure 23

SBox		y															
		00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0a**	**0b**	**0c**	**0d**	**0e**	**0f**
x	**00**	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Table 23

InvSBox		y															
		00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0a**	**0b**	**0c**	**0d**	**0e**	**0f**
x	**00**	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	10	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	20	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	30	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	40	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	50	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	60	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	70	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	80	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	90	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a0	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b0	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c0	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d0	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e0	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f0	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Table 24

While all functions impact the security of the AES block cipher, as the only nonlinear transformation, the S-box plays a crucial role.(Easttom 2014) For this reason, proper S-box design criteria are of the most importance, as it is these design criteria that work to make the AES resistant to linear cryptanalysis, differential cryptanalysis, and algebraic attacks.(Grocholewska-Czurylo 2011) Non-linearity is achieved by ensuring that the maximum input-output correlation amplitude is as small as possible and that the maximum difference propagation probability must be as small as possible.(Daemen & Rijmen 2002) The use of finite field operation in S-Box construction yields linear approximation and difference distribution tables that are close to uniform.(SKABMB 2010) The multiplicative inverse has been determined to be highly non-linear and the substitution is bijective, to ensure invertibility upon decryption.(SKABMB 2010) These properties provide security against differential and linear attacks.(SKABMB 2010)

While the AES has specification for only a single S-Box implementation,(NIST 2001) the Rijndael S-Box allows implementers to select a different configuration,(Daemen & Rijmen 2002) an option that allows a deeper sense of security for those suspicions of a mathematical backdoor built into the cipher.(Katiyar Jeyanthi 2019) The AES is able to provide resistance against differential and linear cryptanalysis if an S-Box with "average" correlation and difference propagation properties is used.(Krishnamurthy Ramaswamy 2008) Modern S-Box' cryptographic strength "is critically analyzed by studying the properties of S-box such as nonlinearity, strict avalanche, bit independence, linear approximation probability and differential approximation probability.(Farwa Shah Idrees 2016 p. 1)

Section 9 : Functions

9.1 : Cipher

This section details a method by which mechanism of the AES cryptographic transformations maybe algorithmically implemented. As a key iterated product cipher, the AES executes a number of round function iterations, on a block of plaintext bits, as discussed in Section 3.3. Each round transformation is executed in the same manner, with variance provided by round values, generally called round constants and a round key. The AES is fundamentally composed of a key schedule and a block cipher. The AES key schedule algorithm calculates round keys and the AES cipher consists of the round function, specifically composed of four byte-oriented transformations. The AES cipher is specifically a key-iterated block cipher, consisting of cipher block transformation rounds, or iterations, defined by the key and block lengths.The number of rounds is determined by:(NIST 2001)

$$Nr = 6 + max(Nb, Nk)$$

Where Nr is the number of round iterations, Nb is the number of 32-bit words in the block, and Nk is the number of 32-bit words in the key.(NIST 2001) Because Nb is fixed to 4 and Nk is at least 4 by the AES specification, the number of rounds Nr, is always determined by the number of bytes in the key Nk.(NIST 2001)

An initial AddRoundKey precedes the iteration of Nr-1 rounds, while the final round does not include the MixColumns() transformation.(NIST 2001) These functions operate on arrays provided by the State and RoundKey pointers.(NIST 2001) RoundKey addressed the key schedule generated by the KeyExpansion() function, a vector consisting of 4-byte words.(NIST 2001) Nb RoundKey words are used each round iteration.(NIST 2001) The State transformations and KeyExpansion() function referenced below are defined in the subsection that follow.

```
Cipher(State,RoundKey)
  begin
    AddRoundKey(State, RoundKey)
    for i = 1 to Nr
      Round(State, RoundKey+Nb*i)
    FinalRound(State, RoundKey+Nb*Nr)
end

Round(State,RoundKey)
  begin
    SubBytes(State);
    ShiftRows(State);
    MixColumns(State);
    AddRoundKey(round, State,RoundKey);
end

FinalRound(State, RoundKey)
  begin
    SubBytes(State) ;
    ShiftRows(State) ;
    AddRoundKey(round, State,RoundKey);
end
```

Figure 24

```
void Cipher(State, RoundKey) {
unsigned char round = 0;
AddRoundKey(0, State, RoundKey);

for (round = 1; round < Nr; ++round) {
  SubBytes(State);
  ShiftRows(State);
  MixColumns(State);
  AddRoundKey(round, State, RoundKey);
  }

  SubBytes(State);
  ShiftRows(State);
  AddRoundKey(Nr, State, RoundKey);
}
```

Figure 25

The AES Cipher() implements a round function composed of four byte-oriented transformations: (NIST 2001)

SubBytes() individual byte substitution using the S-Box lookup table

ShiftRows() shifts each State array rows by a unique offset

MixColumns() mixes each State array column

AddRoundKey() adds the randomized Round Key to the State

9.2 : KeyExpansion

```
KeyExpansion(word cKey, word rKey,
Nk)
begin
  word temp
  for i = 0 step Nk
    rKey[i] = cKey[i];
  end for
  for i = Nk step 1 to Nb*(Nr+1)
    temp = rKey[i-1]
    if (i mod Nk == 0)
      temp = SubWord(RotWord(temp))
             xor Rcon[i/Nk]
    else if (Nk > 6 and i mod Nk == 4)
      temp = SubWord(temp)
    end if
    rKey[i] = rKey[i-Nk] xor temp
    end for
end
```

The function SubWord(x) applies an S-Box substitution to each byte of an input word.(Gladman 2003)

The function RotWord(x) converts an input word $[b_3, b_2, b_1, b_0]$ to an output $[b_0, b_3, b_2, b_1]$.(Gladman 2003)

Figure 26

```
KeyExpansion(byte* rKey, byte* cKey) {
byte r, c, k, tmp, t[4];
for (r=0;r<Nk*4;r++) rKey[r]=cKey[r];
for (r=Nk, k=(r-1)*4; r<4*(Nr+1); r++) {
  t[0]=rKey[k];t[1]=rKey[k+1];
  t[2]=rKey[k+2];t[3]=rKey[k+3];

  if (r % Nk == 0) {
    tmp = Sbox[t[0]];   t[0] = Sbox[t[1]];
    t[1] = Sbox[t[2]];    t[2] = Sbox[t[3]];
    t[3] = tmp;
    t[0]=t[0] ^ Rcon[r / Nk];
    }
  #if defined(AES256)
  if (r%Nk==4){
    t[0]=Sbox[t[0]];t[1]=Sbox[t[1]];
    t[2]=Sbox[t[2]];t[3]=Sbox[t[3]];
    }
  #endif
  c=r*4;   k=(r - Nk)*4;
  rKey[c]=rKey[k] ^ t[0];
  rKey[c+1]=rKey[k+1] ^ t[1];
  rKey[c+2]=rKey[k+2] ^ t[2];
  rKey[c+3]=rKey[k+3] ^ t[3];
  }
}
```

Figure 27

The AES and the Rijndael algorithm, of which the AES is a subset, are classified as Product Ciphers. We introduced the structure of product ciphers in Section 3.3. Product ciphers execute a number of round iterations, each round transformation is executed in the same manner, with variance provided by round values, generally called round constants and a round key.(Van Tilborg & Jajodia, 2011) A key schedule algorithm calculates round keys through the use of simple cryptographic operations, such as S-boxes and P-boxes, on the input cryptographic key.(Van Tilborg & Jajodia, 2011) KeyExspanison() is the AES key schedule.

The AES Cipher is parameterized with a set of Nb input words of key material, and each of the Nr rounds requires Nb words of derived key data.(NIST 2001) Thus, KeyExpansion() results in a total of $Nb(Nr+1)$ words.(NIST 2001) The Key Expansion routine takes the Cipher Key pointer cKey, which addresses the input key material, and the Round Key pointer rKey, which addresses the round key memory. KeyExpansion() uses the rKey pointer to fill the linear array, denoted rKey[i], with i in the range 0 ≤ i < Nb(Nr + 1).

The first Nk words of the $rKey$ are copied from the contents of the $cKey$. Each subsequent word, rKey[i], is equal to the XOR of rKey[i-1] and rKey[i-Nk]. Prior to this XOR, for words of a position that is a multiple of Nk, if(i%Nk==0), a left cyclic shift, an SBox[] table lookup to the word four bytes, and an XOR with a round constant, Rcon[i], is applied to rKey[i-1].(NIST 2001) The round constant word array, Rcon[i], contains the values given by [$x^{(i-1)}$,{00},{00},{00}], with $x^{(i-1)}$ being powers of x in the field GF(2^8).(NIST 2001)

It must also be known that:

- KeyExpansion() for 256-bit Cipher Keys (Nk = 8) is different, if Nk = 8 and i%Nk ==4, for every fourth word, table lookups are applied to rKey[i-1] prior to the XOR.
- The KeyExpansion routine does not need an inverse as the same key material generated for encryption is required by decryption.

9.3 : SubBytes

The $SubBytes()$ transformation is a non-linear byte substitution that operates independently on each State byte.(NIST 2001) As the only non-linear transformation implemented, SubBytes() guarantees the non-linearity of the AES.(NIST 2001)

```
SubBytes(State)
begin
        for i = 0 step 1 4*Nb
                State_i = Sbox[ State_i ]
        end for
end
```

Figure 28

```
static void SubBytes(State* State) {
unsigned char r, c;
for (r = 0; r < 4; ++r) {
  for (c = 0; c < 4; ++c) {
    (*State)[c][r] = Sbox[(*State)[c][r]];
    }
  }
}
```

Figure 29

Each State byte is mapped to a new value by: $b' = Ab^{-1} \oplus c$ (NIST 2001)

This transformation has been previously defined as an AES S-Box transformation of the complete State matrix. As the range of the S-Box is 256, its small size allows specification implementers many options.(Daemen & Rijmen 2002) Commonly, rather than performing the complex operations which produce S-Box values, for each value we can precompute the resultant for storage in an array.(NIST 2001) The AES calls this array a lookup table.(NIST 2001) The S-Box then becomes a square two dimensional lookup table of size 16, containing a permutation of the byte values in GF(28).(NIST 2001) SubBytes() uses the individual contents of a State element as an index into the S-Box substitution table.(NIST 2001) During this transformation the value of each State byte is split into the upper and lower four bits.(NIST 2001) These two new values serve as the S-Box indices.(NIST 2001) The upper four bits determine the S-Box row location and the lower four bits determine the S-Box column location.(NIST 2001) S-Box indices are represented in hex as values of the range $(0-f)$.(NIST 2001) Supposing a State byte $s_{0,0}$ had value $\{11011001\}$, seperate each index (d, 9) and map to the unique S-Box value $\{35\}$ or $\{00110101\}$, illustrated by Figure 30.

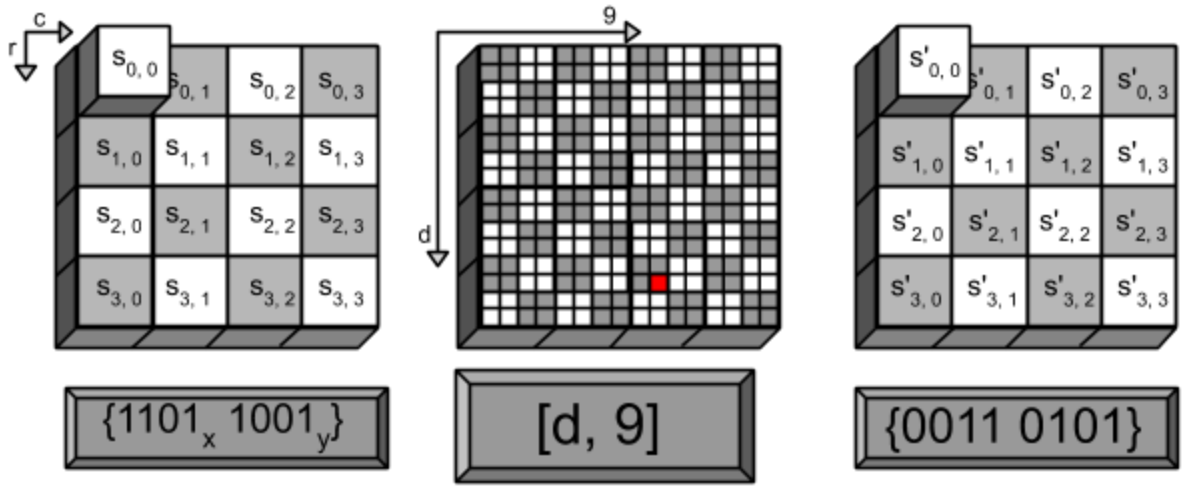

Figure 30

InvSubBytes() is the inverse of SubBytes().[NIST 2001] This means we apply the inverse S-Box to each byte of the State.[NIST 2001] Their mechanism is equivalent, the only difference is the lookup table referenced, S-box by SubBytes() and InvS-box by InvSubBytes(). A full State transformation using SubBytes() then InvSubBytes() is shown by Figure 31 below.

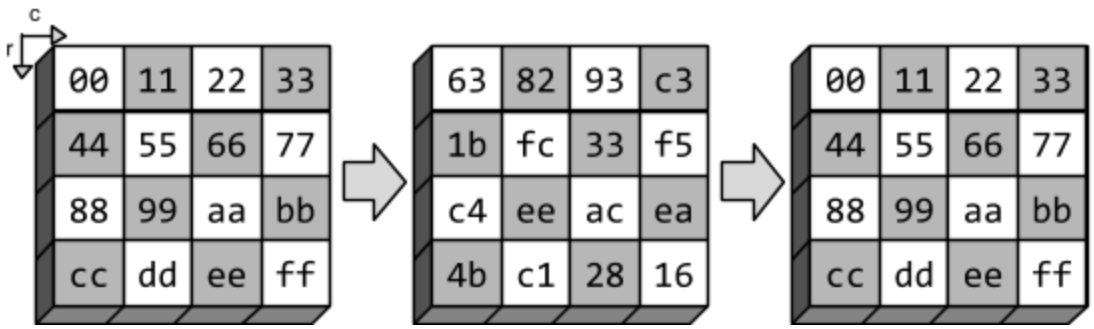

Figure 31

9.4 : ShiftRows

```
ShiftRows(State)
begin
  for r = 0 to 3
    for c = 0 to 3
      State[c] =
        State[r, (c + h(r,Nb)) mod Nb]
    end for
  end for
end
```

Figure 32

```
static void ShiftRows(State* State)
{
  unsigned char temp;

  temp          = (*State)[0][1];
  (*State)[0][1] = (*State)[1][1];
  (*State)[1][1] = (*State)[2][1];
  (*State)[2][1] = (*State)[3][1];
  (*State)[3][1] = temp;

  temp          = (*State)[0][2];
  (*State)[0][2] = (*State)[2][2];
  (*State)[2][2] = temp;

  temp          = (*State)[1][2];
  (*State)[1][2] = (*State)[3][2];
  (*State)[3][2] = temp;


  temp          = (*State)[0][3];
  (*State)[0][3] = (*State)[3][3];
  (*State)[3][3] = (*State)[2][3];
  (*State)[2][3] = (*State)[1][3];
  (*State)[1][3] = temp;
}
```

Figure 33

ShiftRows provides diffusion, performing a transposition on the bytes of each State row.(Daemen & Rijmen 2002) ShiftRows applies a left cyclic shift to each byte in a given row, moving it to the next "lowest" position, with the exception of the byte in the "lowest" position which is shifted out of the state array to "wrap" and appear in the vacant "highest" position.(NIST 2001) The magnitude of each shift is a factor of the row number r, and block length Nb, determined by the relationship:

$$s'_{r,c} = s_{r,(c+\,shift\,(r,\,Nb))\ mod\ Nb}$$

$$for\ 0\ <\ r\ <\ 4\ and\ 0 \leq c\ <\ Nb$$

As the AES specifies $Nb = 4$, coincidentally the shift offset is equal to the row number r, $0 \leq r\ <\ 4$.

The mechanism of this operation is depicted diagrammatically by Figure 34, below.

The four offsets have to be unique for optimal diffusion. Since ShiftRows moves the bytes of each column to four different columns, it is diffusion optimal. "Diffusion optimality is important in providing resistance against differential and linear cryptanalysis."(Daemen & Rijmen 2002 p. 37)

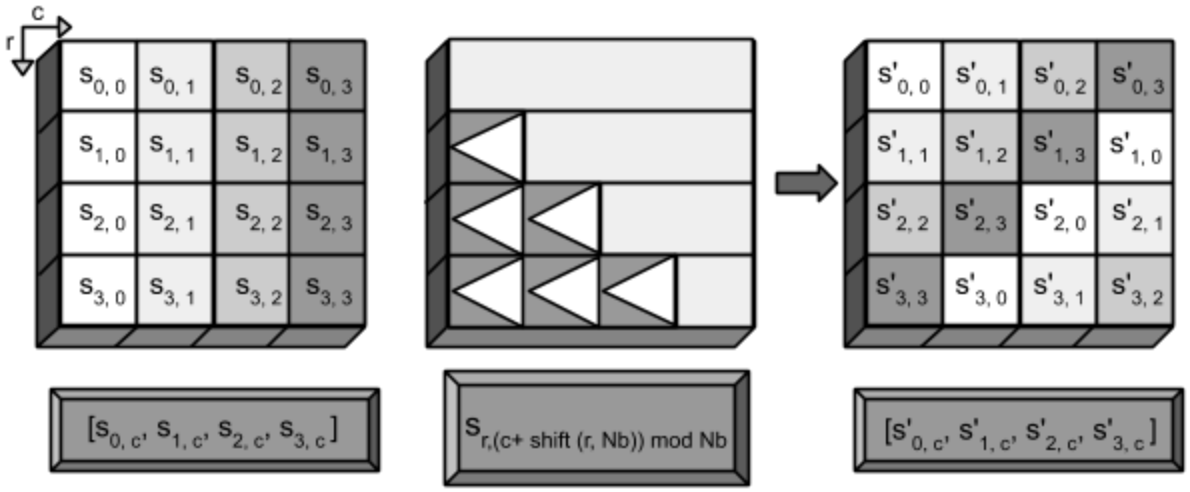

Figure 34

InvShiftRows() is the inverse of the ShiftRows() transformation.(NIST 2001) As before, the first row, $r = 0$, is not shifted while the bytes in the last three State rows, $r = 1, 2, 3$, are cyclically shifted in the manner described before.(NIST 2001) The bottom three rows are shifted by $Nb - shift(r, Nb)$ bytes, where the shift value $shift(r, Nb)$ depends on the row number. This is shown by Figure 35 below, and is given by equation:(NIST 2001)

$$s'_{r,(c+\ shift\ (r,\ Nb))\ mod\ Nb} = s_{r,c} \quad \text{for } 0 < r < 4 \text{ and } 0 \le c < Nb,$$

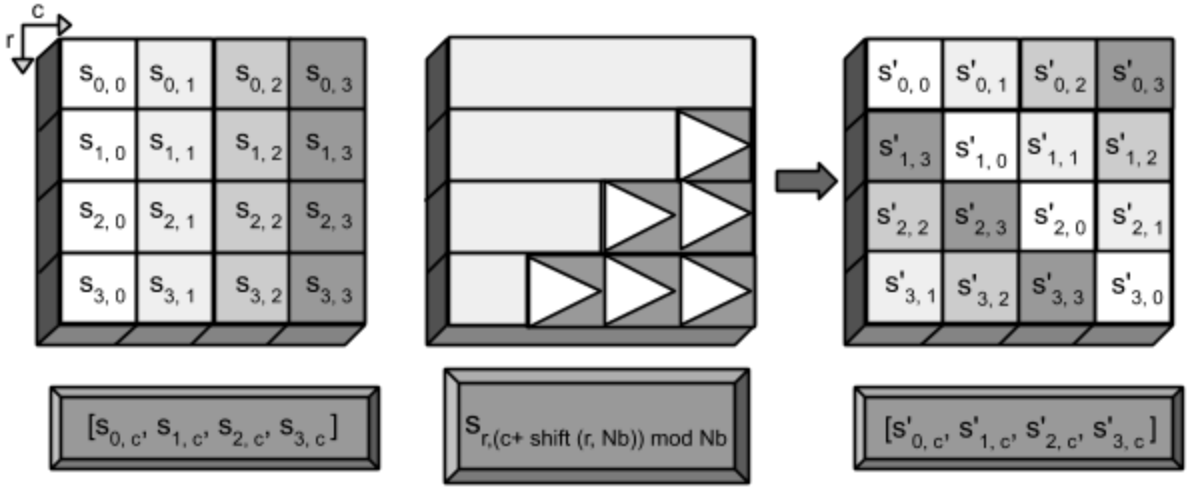

Figure 35

9.5 : MixColumns

```
MixColumns(State)
begin
  byte t[4]
  for c = 0 to 3
    for r = 0 to 3
      t[r] = state[r,c]
    end for
    for r = 0 to 3
      state[r,c] =
      GM(0x02, t[r]) xor
      GM(0x03, t[(r + 1) mod 4]) xor
      t[(r + 2) mod 4] xor
      t[(r + 3) mod 4]
    end for
  end for
end
```

Figure 36

```
xt(a)    ((a&0x80) ? ((a<<1)^0x1b) : (a<<1))

static void MixColumns(State* State) {
unsigned char t0, t1, t2, t3, t, r;

for (r = 0; r < 4; ++r) {

  t0 = State[0] ^ State[1];
  t1 = State[1] ^ State[2];
  t2 = State[2] ^ State[3];
  t3 = State[3] ^ State[0];
  t = t0 ^ t2;

  State[0] ^= xt(t0) ^ t;
  State[1] ^= xt(t1) ^ t;
  State[2] ^= xt(t2) ^ t;
  State[3] ^= xt(t3) ^ t;
        }
}
```

Figure 37

The $MixColumns()$ transformation sequentially processes each of the four State columns denoted $\{c_0, c_1, c_2, c_3\}$.(NIST 2001) $MixColumns()$ operates on each of the four 4-byte word columns which encompass all sixteen AES state values.(NIST 2001) Due to the mechanism of this transformation, each byte in the input affects all four bytes of the output, such that $MixColumns()$ provides diffusion.(Daemen & Rijmen 2002)

The $MixColumns()$ transformation applies Modular Multiplication in Rijndael's Galois field.(NIST 2001) This mechanism is described in full by Section 7.9. Each column is a word, represented via

four-term polynomial with coefficients over GF(2^8). Under the AES, polynomials over GF(2^8) are multiplied by a fixed polynomial $a(x) = 3x^3 + x^2 + x + 2$ modulo $x^4 + 1$.[NIST 2001]

Figure 38, below, shows how this can be written as the matrix multiplication $s'(x) = a(x) \otimes s(x)$:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Figure 38

An equivalent system of equations achieves the result of this multiplication shown by Figure 39, below, the four bytes in a column are solved for by the following:[NIST 2001]

$$\begin{aligned} s'_{0,c} &= (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c} \\ s'_{1,c} &= s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c} \\ s'_{2,c} &= s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c}) \\ s'_{3,c} &= (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c}). \end{aligned}$$

Figure 39

All AES operations are done in the Galois field GF(2^8). Addition is an exclusive or (XOR) operation as explained in Section 7.7 which we will use the symbol $\oplus$ to represent. Multiplication is a complex operation as defined in Section 7.8 which we will use the symbol • to represent. For the sake of explanation we temporarily use GM() as an abstraction for a GF(2^8) multiplication function. GM() takes two GF(2^8) field members and returns their product under AES GF(2^8) multiplication. Thus, GM() usage below implements the GF(2^8) multiplication defined in Section 7.8. Furthermore, c defines a column position counter, {0,1,2,3}. s defines a four byte array representing the state column before the AES MixColumns transformation. with byte values represented by s[c] = {s[0],s[1],s[2],s[3]}. s' defines a four byte array representing the state column after the AES MixColumns transformation. with byte values represented by s'[c] = {s'[0],s'[1],s'[2],s'[3]}.

The state array can be arranged in memory such that each state column is a four byte, 32-bit word, c[3] to c[0]. The mixColumns transformation is then: (Gladman 2003 p. 16)

$c[3]' = \{02\} \bullet c[3] \oplus \{03\} \bullet c[0] \oplus c[1] \oplus c[2]$

$c[2]' = \{02\} \bullet c[2] \oplus \{03\} \bullet c[3] \oplus c[0] \oplus c[1]$

$c[1]' = \{02\} \bullet c[1] \oplus \{03\} \bullet c[2] \oplus c[3] \oplus c[0]$

$c[0]' = \{02\} \bullet c[0] \oplus \{03\} \bullet c[1] \oplus c[2] \oplus c[3]$

By our notation:

s'[0]=GM(s[0],2)⊕GM(s[3],1)⊕GM(s[2],1)⊕GM(s[1],3);

s'[1]=GM(s[1],2)⊕GM(s[0],1)⊕GM(s[3],1)⊕GM(s[2],3);

s'[2]=GM(s[2],2)⊕GM(s[1],1)⊕GM(s[0],1)⊕GM(s[3],3);

s'[3]=GM(s[3],2)⊕GM(s[2],1)⊕GM(s[1],1)⊕GM(s[0],3);

Simplify by $GM(s[n], 1) = s[n]$:

Multiplication by 1 is multiplication by the identity element which leave a member unchanged.

s'[0] = GM(s[0],2) ⊕ s[3] ⊕ s[2] ⊕ GM(s[1],3);

s'[1] = GM(s[1],2) ⊕ s[0] ⊕ s[3] ⊕ GM(s[2],3);

s'[2] = GM(s[2],2) ⊕ s[1] ⊕ s[0] ⊕ GM(s[3],3);

s'[3] = GM(s[3],2) ⊕ s[2] ⊕ s[1] ⊕ GM(s[0],3);

Simplify by $GM(s[n], 3) = GM(s[n], 2) \oplus s[n]$:

Multiplication by 3 is equivalent to Multiplication by 2 and an XOR,

s'[0] = GM(s[0],2) ⊕ s[3] ⊕ s[2] ⊕ GM(s[1],2)⊕s[1];

s'[1] = GM(s[1],2) ⊕ s[0] ⊕ s[3] ⊕ GM(s[2],2)⊕s[2];

s'[2] = GM(s[2],2) ⊕ s[1] ⊕ s[0] ⊕ GM(s[3],2)⊕s[3];

s'[3] = GM(s[3],2) ⊕ s[2] ⊕ s[1] ⊕ GM(s[0],2)⊕s[0]

Simplify by $GM(x, 2) \oplus GM(y, 2) = GM(x \oplus y, 2)$:

Due to the associative and distributive properties, we perform the XOR of GM parameters first, to reduced GM calls by half.

s'[0] = GM(s[0]⊕ s[1],2) ⊕ s[3] ⊕ s[2] ⊕ s[1];

s'[1] = GM(s[1]⊕ s[2],2) ⊕ s[0] ⊕ s[3] ⊕ s[2];

s'[2] = GM(s[2]⊕ s[3],2) ⊕ s[1] ⊕ s[0] ⊕ s[3];

s'[3] = GM(s[3]⊕ s[0],2) ⊕ s[2] ⊕ s[1] ⊕ s[0];

Additionally, Multiplication by 2 in GF(2^8) is multiplication by x (binary {00000010} or hexadecimal {02}).(NIST 2001) We now explain how multiplication by x is implemented as a left shift and a subsequent conditional bitwise XOR with {1b}.(NIST 2001) Multiplying binary polynomials b(x) and x produces: $b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x$.(NIST 2001) The result x • b(x) in GF(2^8) is

then obtained by reducing the above result modulo m(x), see Section 7.8.[NIST 2001] To determine if such reduction is necessary we perform the following conditional check. If b7 = 0, the result is already in reduced form, If b7 = 1, we perform reduction by XOR of the polynomial m(x).[NIST 2001]

multiplication by x, implemented by left shift. $b[c] = s[c] << 1;$

A bitmask is used to determine if b_7 =1

If so , we perform reduction by m(x)

$$if(s[c] \,\&\, 0X80)$$
$$b[r] \oplus = 0X1b;$$

Simplify by $GM(s[n], 2) = xt(s[n])$:

This is the AES operation xtime(),[NIST 2001]

for which we #define xt(x)

((a & 0X80) ? ((a << 1) ⊕ 0X1b) : (a << 1))

s'[0] = xt(s[0]⊕ s[1],2) ⊕ s[3] ⊕ s[2] ⊕ s[1];
s'[1] = xt(s[1]⊕ s[2],2) ⊕ s[0] ⊕ s[3] ⊕ s[2];
s'[2] = xt(s[2]⊕ s[3],2) ⊕ s[1] ⊕ s[0] ⊕ s[3];
s'[3] = xt(s[3]⊕ s[0],2) ⊕ s[2] ⊕ s[1] ⊕ s[0];

temporary values reduce load/arithmetic ops

t0 = s[0] ⊕ s[1]; t2 = s[2] ⊕ s[3];

t1 = s[1] ⊕ s[2]; t3 = s[3] ⊕ s[0];

t = t0 ⊕ t1;

s'[0] = s[0] ⊕ xt(t0) ⊕ t;
s'[1] = s[1] ⊕ xt(t1) ⊕ t;
s'[2] = s[2] ⊕ xt(t2) ⊕ t;
s'[3] = s[3] ⊕ xt(t3) ⊕ t;

9.6 : InvMixColumns

```
InvMixColumns(State)
begin
  byte t[4]
  for c = 0 to 3
    for r = 0 to 3
      t[r] = state[r,c]
    end for
    for r = 0 to 3
      state[r,c] =
      GM(0x0E, t[r]) xor
      GM(0x0B, t[(r + 1) mod 4]) xor
      GM(0x0D, t[(r + 2) mod 4]) xor
      GM(0x09, t[(r + 3) mod 4]) xor
    end for
  end for
end
```

Figure 40

```
static void InvMixColumns(State* State) {
unsigned char r, t0, t1, t2, t3;

for (r = 0; r < 4; ++r) {
  t0 = State[0]; t1 = State[1];
  t2 = State[2]; t3 = State[3];

  State[0]=GM_E[t0]^GM_9[t3]^GM_D[t2]^GM_B[t1];
  State[1]=GM_E[t1]^GM_9[t0]^GM_D[t3]^GM_B[t2];
  State[2]=GM_E[t2]^GM_9[t1]^GM_D[t0]^GM_B[t3];
  State[3]=GM_E[t3]^GM_9[t2]^GM_D[t1]^GM_B[t0];
  }
}
```

Figure 41

The InvMixColumns() transformation sequentially processes each of the four State columns denoted $\{c_0, c_1, c_2, c_3\}$. InvMixColumns operates on each of the four, 4-byte word columns to encompass all sixteen AES state values. The InvMixColumns transformation applies Modular Multiplication in Rijndael's Galois field. This mechanism is described in full by Section 7.9. Each column is a word, represented via four-term polynomial with coefficients over GF(2^8). Under the AES, polynomials over GF(2^8) are multiplied by fixed polynomial $a^{-1}(x) = \{0B\}x^3 + \{0D\}x^2 + \{09\}x + \{0E\}$ modulo $x^4 + 1$.(NIST 2001)

Figure 42, below, shows how this can be written as the matrix multiplication $s'(x) = a(x) \otimes s(x)$.(NIST 2001)

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Figure 42

An equivalent system of equations achieves the result of this multiplication shown by Figure 43, below, the four bytes in a column are solved for by the following:(NIST 2001)

$$s'_{0,c} = (\{0e\} \bullet s_{0,c}) \oplus (\{0b\} \bullet s_{1,c}) \oplus (\{0d\} \bullet s_{2,c}) \oplus (\{09\} \bullet s_{3,c})$$
$$s'_{1,c} = (\{09\} \bullet s_{0,c}) \oplus (\{0e\} \bullet s_{1,c}) \oplus (\{0b\} \bullet s_{2,c}) \oplus (\{0d\} \bullet s_{3,c})$$
$$s'_{2,c} = (\{0d\} \bullet s_{0,c}) \oplus (\{09\} \bullet s_{1,c}) \oplus (\{0e\} \bullet s_{2,c}) \oplus (\{0b\} \bullet s_{3,c})$$
$$s'_{3,c} = (\{0b\} \bullet s_{0,c}) \oplus (\{0d\} \bullet s_{1,c}) \oplus (\{09\} \bullet s_{2,c}) \oplus (\{0e\} \bullet s_{3,c})$$

Figure 43

Rather than perform Galois Field multiplication directly, AES implementations have previously relied upon lookup tables to perform GF(2^8) multiplication. Necessary for the transformations of the AES by this method are six identical tables each consisting of all values in GF(2^8), stored in 256 byte arrays. The members of these tables are then multiplied by a given constant in GF(2^8). Multiplication by the constants 2 and 3, are required for encryption. Multiplication by the constants 9, 11, 13, and 14 are required for decryption. This particular implementation shows decryption by table. Required are the values shown below by Tables 25, 26, 27, 28, which depicts multiplication of the constants 9, 11, 13, and 14, respectively. For example, to achieve multiplication of the value n by 9 in GF(2^8) via a table array, the result r, is generally obtained:
$byte\ result\ =\ GM9[n];$

$GM\_9[n]$		y															
		00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0a**	**0b**	**0c**	**0d**	**0e**	**0f**
x	**00**	00	09	12	1b	24	2d	36	3f	48	41	5a	53	6c	65	7e	77
	10	90	99	82	8b	b4	bd	a6	af	d8	d1	ca	c3	fc	f5	ee	e7
	20	3b	32	29	20	1f	16	0d	04	73	7a	61	68	57	5e	45	4c
	30	ab	a2	b9	b0	8f	86	9d	94	e3	ea	f1	f8	c7	ce	d5	dc
	40	76	7f	64	6d	52	5b	40	49	3e	37	2c	25	1a	13	08	01
	50	e6	ef	f4	fd	c2	cb	d0	d9	ae	a7	bc	b5	8a	83	98	91
	60	4d	44	5f	56	69	60	7b	72	05	0c	17	1e	21	28	33	3a
	70	dd	d4	cf	c6	f9	f0	eb	e2	95	9c	87	8e	b1	b8	a3	aa
	80	ec	e5	fe	f7	c8	c1	da	d3	a4	ad	b6	bf	80	89	92	9b
	90	7c	75	6e	67	58	51	4a	43	34	3d	26	2f	10	19	02	0b
	a0	d7	de	c5	cc	f3	fa	e1	e8	9f	96	8d	84	bb	b2	a9	a0
	b0	47	4e	55	5c	63	6a	71	78	0f	06	1d	14	2b	22	39	30
	c0	9a	93	88	81	be	b7	ac	a5	d2	db	c0	c9	f6	ff	e4	ed
	d0	0a	03	18	11	2e	27	3c	35	42	4b	50	59	66	6f	74	7d
	e0	a1	a8	b3	ba	85	8c	97	9e	e9	e0	fb	f2	cd	c4	df	d6
	f0	31	38	23	2a	15	1c	07	0e	79	70	6b	62	5d	54	4f	46

Table 25

$GM\_B[n]$		y															
		00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0a**	**0b**	**0c**	**0d**	**0e**	**0f**
x	**00**	00	0b	16	1d	2c	27	3a	31	58	53	4e	45	74	7f	62	69
	10	b0	bb	a6	ad	9c	97	8a	81	e8	e3	fe	f5	c4	cf	d2	d9
	20	7b	70	6d	66	57	5c	41	4a	23	28	35	3e	0f	04	19	12
	30	cb	c0	dd	d6	e7	ec	f1	fa	93	98	85	8e	bf	b4	a9	a2
	40	f6	fd	e0	eb	da	d1	cc	c7	ae	a5	b8	b3	82	89	94	9f
	50	46	4d	50	5b	6a	61	7c	77	1e	15	08	03	32	39	24	2f
	60	8d	86	9b	90	a1	aa	b7	bc	d5	de	c3	c8	f9	f2	ef	e4
	70	3d	36	2b	20	11	1a	07	0c	65	6e	73	78	49	42	5f	54
	80	f7	fc	e1	ea	db	d0	cd	c6	af	a4	b9	b2	83	88	95	9e
	90	47	4c	51	5a	6b	60	7d	76	1f	14	09	02	33	38	25	2e
	a0	8c	87	9a	91	a0	ab	b6	bd	d4	df	c2	c9	f8	f3	ee	e5
	b0	3c	37	2a	21	10	1b	06	0d	64	6f	72	79	48	43	5e	55
	c0	01	0a	17	1c	2d	26	3b	30	59	52	4f	44	75	7e	63	68
	d0	b1	ba	a7	ac	9d	96	8b	80	e9	e2	ff	f4	c5	ce	d3	d8
	e0	7a	71	6c	67	56	5d	40	4b	22	29	34	3f	0e	05	18	13
	f0	ca	c1	dc	d7	e6	ed	f0	fb	92	99	84	8f	be	b5	a8	a3

Table 26

$GM\_D[n]$		y															
		00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0a**	**0b**	**0c**	**0d**	**0e**	**0f**
x	**00**	00	0d	1a	17	34	39	2e	23	68	65	72	7f	5c	51	46	4b
	10	d0	dd	ca	c7	e4	e9	fe	f3	b8	b5	a2	af	8c	81	96	9b
	20	bb	b6	a1	ac	8f	82	95	98	d3	de	c9	c4	e7	ea	fd	f0
	30	6b	66	71	7c	5f	52	45	48	03	0e	19	14	37	3a	2d	20
	40	6d	60	77	7a	59	54	43	4e	05	08	1f	12	31	3c	2b	26
	50	bd	b0	a7	aa	89	84	93	9e	d5	d8	cf	c2	e1	ec	fb	f6
	60	d6	db	cc	c1	e2	ef	f8	f5	be	b3	a4	a9	8a	87	90	9d
	70	06	0b	1c	11	32	3f	28	25	6e	63	74	79	5a	57	40	4d
	80	da	d7	c0	cd	ee	e3	f4	f9	b2	bf	a8	a5	86	8b	9c	91
	90	0a	07	10	1d	3e	33	24	29	62	6f	78	75	56	5b	4c	41
	a0	61	6c	7b	76	55	58	4f	42	09	04	13	1e	3d	30	27	2a
	b0	b1	bc	ab	a6	85	88	9f	92	d9	d4	c3	ce	ed	e0	f7	fa
	c0	b7	ba	ad	a0	83	8e	99	94	df	d2	c5	c8	eb	e6	f1	fc
	d0	67	6a	7d	70	53	5e	49	44	0f	02	15	18	3b	36	21	2c
	e0	0c	01	16	1b	38	35	22	2f	64	69	7e	73	50	5d	4a	47
	f0	dc	d1	c6	cb	e8	e5	f2	ff	b4	b9	ae	a3	80	8d	9a	97

Table 27

$GM\_E[n]$		y															
		00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0a**	**0b**	**0c**	**0d**	**0e**	**0f**
x	**00**	00	0e	1c	12	38	36	24	2a	70	7e	6c	62	48	46	54	5a
	10	e0	ee	fc	f2	d8	d6	c4	ca	90	9e	8c	82	a8	a6	b4	ba
	20	db	d5	c7	c9	e3	ed	ff	f1	ab	a5	b7	b9	93	9d	8f	81
	30	3b	35	27	29	03	0d	1f	11	4b	45	57	59	73	7d	6f	61
	40	ad	a3	b1	bf	95	9b	89	87	dd	d3	c1	cf	e5	eb	f9	f7
	50	4d	43	51	5f	75	7b	69	67	3d	33	21	2f	05	0b	19	17
	60	76	78	6a	64	4e	40	52	5c	06	08	1a	14	3e	30	22	2c
	70	96	98	8a	84	ae	a0	b2	bc	e6	e8	fa	f4	de	d0	c2	cc
	80	41	4f	5d	53	79	77	65	6b	31	3f	2d	23	09	07	15	1b
	90	a1	af	bd	b3	99	97	85	8b	d1	df	cd	c3	e9	e7	f5	fb
	a0	9a	94	86	88	a2	ac	be	b0	ea	e4	f6	f8	d2	dc	ce	c0
	b0	7a	74	66	68	42	4c	5e	50	0a	04	16	18	32	3c	2e	20
	c0	ec	e2	f0	fe	d4	da	c8	c6	9c	92	80	8e	a4	aa	b8	b6
	d0	0c	02	10	1e	34	3a	28	26	7c	72	60	6e	44	4a	58	56
	e0	37	39	2b	25	0f	01	13	1d	47	49	5b	55	7f	71	63	6d
	f0	d7	d9	cb	c5	ef	e1	f3	fd	a7	a9	bb	b5	9f	91	83	8d

Table 28

9.7 : AddRoundKey

```
AddRoundKey(State, RoundKey)
begin
        for i = 0 step 1 4*Nb
                State_i XOR RoundKey_i
        end for
end
```

Figure 44

```
static void AddRoundKey(unsigned char round,
State* State,unsigned char* RoundKey) {
unsigned char r, c;
  for (r = 0; r < 4; ++r) {
    for (c = 0; c < 4; ++c) {
      (*State)[r][c] ^=
        RoundKey[(round*Nb*4) + (r*Nb) + c];
    }
  }
}
```

Figure 45

The AddRoundKey function performs a bitwise exclusive or between the State and a Round Key.[NIST 2001] Each Round Key is a portion of the key schedule, generated by the KeyExpansion() function, a 4-byte word vector.[NIST 2001] A Round Key consists of Nb words, where Nb is the number of words in a cipher block, $Nb = 4$ by the specification of the AES. Round Key addition occurs prior to the first round, and continues when 1 ≤ round ≤ Nr.[NIST 2001] Each word of the Round Key is XOR'ed into the columns of the State:

$$\{ s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c} \} \oplus w_{round*Nb+c} = \{ s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c} \} \qquad for\ 0 \le c < Nb \quad and \quad 0 \le round \le Nr$$

Where $s_{r,c}$ represents a State byte, w_i represents a given key schedule word and $s'_{r,c}$ is a transformed byte. The mechanism of this operation is depicted diagrammatically by Figure 46, below. AddRoundKey(), described above, is equivalent to the XOR operation and is thus its own inverse.[NIST 2001]

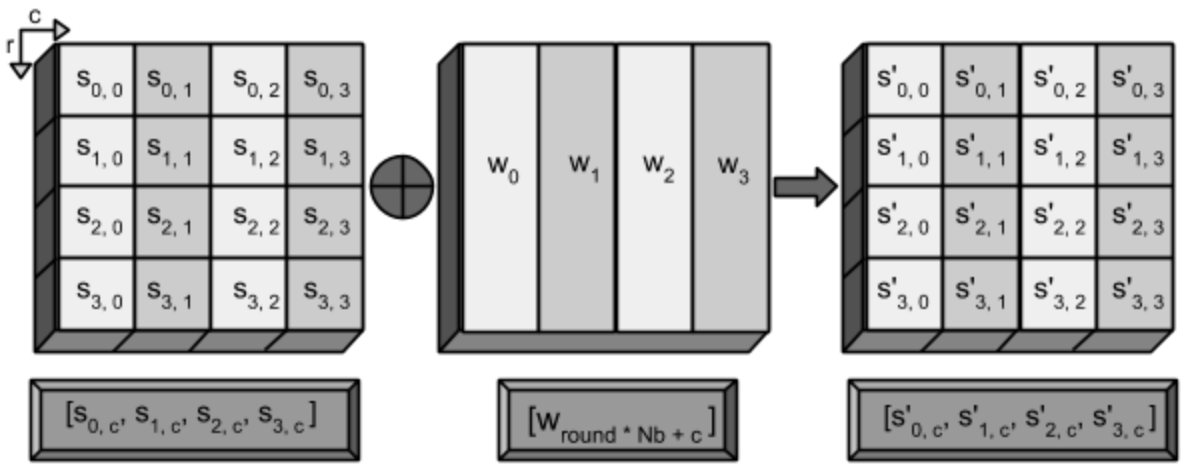
c
r
s0, 0 s0, 1 s0, 2 s0, 3
s1, 0 s1, 1 s1, 2 s1, 3
s2, 0 s2, 1 s2, 2 s2, 3
s3, 0 s3, 1 s3, 2 s3, 3
w0 w1 w2 w3
s'0, 0 s'0, 1 s'0, 2 s'0, 3
s'1, 0 s'1, 1 s'1, 2 s'1, 3
s'2, 0 s'2, 1 s'2, 2 s'2, 3
s'3, 0 s'3, 1 s'3, 2 s'3, 3
[s0, c, s1, c, s2, c, s3, c]
[wround * Nb + c]
[s'0, c, s'1, c, s'2, c, s'3, c]

Figure 46

9.8 : InvCipher

Transformations of the AES Cipher() are decrypted through round and function inversion by InvCipher(). The same number of round iterations are used.[NIST 2001]

```
InvCipher(State,RoundKey)
begin
  AddRoundKey(State, RoundKey)
  for i = 1 to Nr
    InvRound(State, RoundKey+Nb*i);
    InvFinalRound(State, RoundKey+Nb*Nr);
  end for

InvRound(State,RoundKey)
begin
  InvShiftRows(State);
  InvSubBytes(State);
  AddRoundKey(round, State, RoundKey);
  InvMixColumns(State);
end

InvFinalRound(State, RoundKey)
begin
  InvShiftRows(State);
  InvSubBytes(State);
  AddRoundKey(round, State, RoundKey);
end
```

Figure 47

```
void InvCipher(State* State, unsigned char*
RoundKey)
{
unsigned char round = 0;

AddRoundKey(Nr, State, RoundKey);

for (round = (Nr - 1); round > 0; round--) {
  InvShiftRows(State);
  InvSubBytes(State);
  AddRoundKey(round, State, RoundKey);
  InvMixColumns(State);
  }

InvShiftRows(State);
InvSubBytes(State);
AddRoundKey(0, State, RoundKey);
}
```

Figure 48

The AES InvCipher() implements a round function of four byte-oriented transformations:[(NIST 2001)]

InvSubBytes() individual byte substitution using the InvS-Box lookup table

InvShiftRows() shifts each State array rows by a unique offset

InvMixColumns() mixes each State array column

AddRoundKey() adds the randomized Round Key to the State

An initial AddRoundKey precedes the iteration of Nr-1 rounds, while the final round does not include the InvMixColumns() transformation. These functions operate on arrays provided by the State and RoundKey pointers. RoundKey is the key schedule generated by the KeyExpansion() function, a vector consisting of 4-byte words. Nb RoundKey words are used each round iteration. The State transformations referenced above are defined in the same sections as their inverses except for InvMixColumns which is Section 9.6.

Section 10 : Block Cipher Modes of Operation
10.1 : Probabilistic Encryption

Individually a block cipher serves to provide message confidentiality, protection from unauthorized access.(Paar & Pelzi 2009) Unauthorized access in this case equates to adversaries without the block ciphers secret key.(Paar & Pelzi 2009) However, a block cipher is only defined for a single block size transformation per key. In practice, the size of a message is larger than the block size, often much larger. Secure use of a block cipher would entail impractical key generation and management efforts. It is because of this that block ciphers are classified as cryptographic primitives, to be used as a component in a secure cryptosystem.(Van Tilborg & Jajodia, 2011)

Standardised block cipher modes of operation were developed to extend block cipher capability. While their applications are many, the few focused by this document are procedures that allow a generic block cipher to transform data allocations larger than a single block and achieve probabilistic results under a fixed key.(Van Tilborg & Jajodia, 2011) Of primary concern when encrypting with a constant key, is the tendency for equal plaintext blocks to share some non-random correspondence.(Van Tilborg & Jajodia, 2011) To be semantically secure, protecting all plaintext information, an encryption algorithm's execution must be probabilistic.(Housley 2004) While we discussed deterministic execution in Section 2.2, a probabilistic encryption method is defined as a process that introduces randomness to every instantiation. In the case of block cipher algorithms, probabilistic encryption ensures that identical plaintexts under a constant key result in unique ciphertexts, masking data patterns. An example of these patterns is illustrated.(Housley 2004)

To achieve probabilistic encryption we must provide randomness for each block cipher instantiation. As input, along with the message and key, most modes require an **initialization vector (IV)**.(Housley 2004) An IV is a block-sized bit vector, comprising a unique binary sequence, used to randomize encryption under a constant key.(Kuo-Tsang Huang Chiu Shen 2013) When implemented in an established mode of operation, IVs eliminate the need for a slower re-keying process.(Kuo-Tsang Huang Chiu Shen 2013) As they serve a different purpose, Initialization vectors have different security requirements than keys. Most apparent is the fact that IVs do not need to be secret.(Kuo-Tsang Huang Chiu Shen 2013) Of greatest importance is that IV's must not be used reused with the same key, and, for some modes, IV generation must be unpredictable.(Kuo-Tsang Huang Chiu Shen 2013)

Additionally, some modes increase block cipher capability to provide properties which complement the security of the underlying block cipher. A common requirement are forms of encryption which assure message confidentiality and authenticity.(Dworkin 2001) Message

authenticity is a property held by messages for which a receiver can verify unmodified transmission from a known origin.[Dworkin 2001] Many modes of operation have been defined, as such their security qualities and use cases vary. NIST has defined five modes of operation for AES and other FIPS-approved block ciphers. Each of these modes has different characteristics. The five modes are: ECB (Electronic Code Book), CBC (Cipher Block Chaining), CFB (Cipher FeedBack), OFB (Output FeedBack), and CTR (Counter).[Housley 2004] Care must be taken during implementation and application to maintain security or complete compromise is risked.[Dworkin 2001] This document uses ECB, CBC, and CTR to serve as examples of operation mode variety,

10.2 : Padding

Block cipher primitives are defined to operate on fixed size plaintext blocks.[(Dworkin 2001)] While block ciphers may be capable of operating on blocks of varied size, block size is a constant during individual execution instances.[(Dworkin 2001)] Plaintext that is not a multiple of the current block size must be padded.[(Menezes Van Oorshot & Vanstone 1997)] Padding is the addition of plaintext material such that message size is a multiple of the cipher block size,[(Menezes Van Oorshot & Vanstone 1997)] many methods exist.[(Ferguson Schneier 2003)] Two of the examples outlined by this document, ECB and CBC, require that final blocks be padded before encryption.[(Dworkin 2001)] We will explore a few padding methods of trivial complexity below.

The simplest padding methods append a bytes to the length of the plaintext p, such that their total is b, the cipher block size $p + a = xb$. These a bytes have designated values such that their removal is procedural. Zero padding appends a {00} byte values:

Padding an 8 byte block: ... | PP PP PP PP PP PP PP PP | PP PP PP PP **00 00 00 00** |

Zero padding is unusable on plaintexts ending in one or more zero byte values. The boundary between the plaintext and the pad string is ambiguous as "trailing 0-bits of the original data cannot be distinguished from those added during padding."[(Menezes Van Oorshot & Vanstone 1997 p. 335)] The {00} value in Zero padding can be replaced with any byte value. This method is acceptable, and efficient, if recipients are able to know the message length.[(Menezes Van Oorshot & Vanstone 1997)]

The method is similar in concept, but solves the boundary issue. A predefined byte value is appended to mark the plaintext boundary, then the remaining $(a - 1)$ bytes are zero padded. ISO/IEC 7816-4:2005 defines this method to be used for 8 byte smart cards when the boundary byte is 0X80.

Padding an 8 byte block: ... | PP PP PP PP PP PP PP PP | PP PP PP PP **80 00 00 00** |

If the message length is a multiple of the cipher block size, an entire padding block with be appended to ensure the plaintext boundary can be determined.[(Menezes Van Oorshot & Vanstone 1997)] This is due to cases where the plaintext to be padded is the block size and the final plaintext byte is equal to the value of the boundary value.[(Menezes Van Oorshot & Vanstone 1997)] Without appending an entire block, the final byte is just as likely to be a padding boundary byte as it is to be valid

plaintext.(Menezes Van Oorshot & Vanstone 1997) Bytes to be stripped from a decrypted plaintext are the boundary value and any {00} bytes that follow.

Our final example has been standardized for cryptographically protected messages. This padding method is described in RFC 5652, Cryptographic Message Syntax (CMS), Section 6.3. CMS has been thoroughly reviewed and is approved for used to digitally sign, digest, authenticate, or encrypt arbitrary message content.(Housley 2009) The padding string consists of a bytes of value a, where $0 \leq a < 256$, This method is only well defined for values that can be expressed by a byte.(Housley 2009)

Padding an 8 byte block: ... | PP PP PP PP PP PP PP PP | PP PP PP PP **04 04 04 04** |

As with the previous method, if the message is a multiple of the block size, an entire plaintext block will be appended to provide a defined plaintext boundary. The number of bytes to be stripped from decrypted plaintext is equal to the value of the final byte.(Housley 2009)

Modes that require padding can compromise the security benefits provided by the underlying block cipher. The addition of a padding transformation introduces the possibility of attacks where by adversaries use information leaked about the padding process to compromise the underlying block cipher.(Fedler 2013) The description of such a process is outside the scope of this document. Contemporary padding procedures do not require padding as ciphertext is made equal to the message in size with negligible complexity increase. Methods like ciphertext stealing, which provides an alternative specification to the popular CBC mode which are not detailed by this document.(Kuo-Tsang Huang Chiu Shen 2013)

A security measure within the scope of this document details the use of a "streaming" mode where by the plaintext length does not need to be a multiple of the blocksize, the example outlined by this document is CTR.(Dworkin 2001) Streaming block cipher modes, like stream ciphers themselves, do not require padding.(Dworkin 2001) Such modes use the block cipher to generate a stream of pseudo random data to be xored with the plaintext.(Dworkin 2001) Similar to the One Time Pad, the random data required is equal to the size of the message, rather than a multiple of the block size. Because of this, streaming modes are used in applications where it is inefficient to add padding.

10.3 : Electronic Codebook (ECB)

ECB is the most simplistic standardized encryption mode.(Van Tilborg & Jajodia, 2011) A message of size m is divided into a number of size n plaintext blocks, where n is the cipher block size. Each plaintext block p_i, is encrypted with a constant key k, to produce ciphertext block c_i.(Paar & Pelzi 2009) ECB mode is its own inverse.(Dworkin 2001)

Execution

- $E(p_0, k) = c_0$ A plaintext block p_0, is encrypted with the key k, to produce ciphertext block c_0
- $E(p_i, k) = c_i$ Subsequent blocks p_i, are encrypted with key k to produce ciphertext blocks c_i

Where i is the block index $0 \leq i < q$, and q is the number of size n plaintext blocks such that $qn = m$.(Paar & Pelzi 2009) As there is no interaction between cipher execution on individual blocks, thus, there is no dependence between the transformation of a given message block and any subsequent operations.(Housley 2004) Due to this lack of dependence, ECB is possible to implement in parallel and transcription errors affect only the containing block.(Paar & Pelzi 2009) We will see that block cipher modes with dependence between block transformations are impossible to implement in parallel and tend to cascade transcription errors such that a fault in one bit may invalidate all subsequent transformations.(Paar & Pelzi 2009)

Given a particular input, a deterministic procedure will always execute the same sequence of states, producing identical output.(Van Tilborg & Jajodia, 2011) By ECB, if two equal plaintext blocks are encrypted under the same key, the corresponding cipher operations and ciphertext blocks will be identical, thusly ECB mode is deterministic.(Housley 2004) The mode's name originates from the fact that, for a given key, a codebook could be created, mapping all possible of ciphertexts for all possible plaintext blocks.(Dworkin 2001) This reduces block transformations to a table lookup, ", analogous to the assignment of code words in a codebook"(Dworkin 2001 p. 9), the type used for decades by the financial industry.(Paar & Pelzi 2009)

In general, deterministic ciphers are considered insecure, ECB mode should not be used without thorough consideration.(Housley 2004) This insecurity is due to a lack of diffusion, data patterns of the underlying message can be seen in ECB mode output, resulting from identical plaintext blocks corresponding to identical ciphertext blocks.(Housley 2004) A common example is the use of ECB to encrypt an uncompressed bitmap image,(Kuo-Tsang Huang Chiu Shen 2013) as seen in

the left most image of Figure 49 below. Bitmap pixels are represented by a defined byte pattern, areas with equal pixel values result in equal pixel values upon ECB encryption as represented in the middle image of Figure 50 below.(Kuo-Tsang Huang Chiu Shen 2013) In this sense ECB does not achieve basic message confidentiality.(Kuo-Tsang Huang Chiu Shen 2013) When a probabilistic encryption mode is used, encryption of the equal pixel values result in distinct pixel values after encryption as seen in the right most image of Figure 51 below.(Kuo-Tsang Huang Chiu Shen 2013)

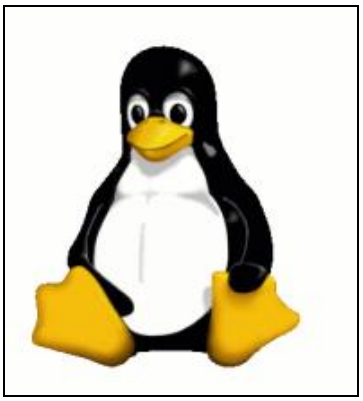

Figure 49

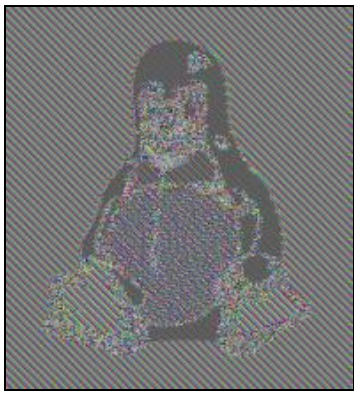

Figure 50

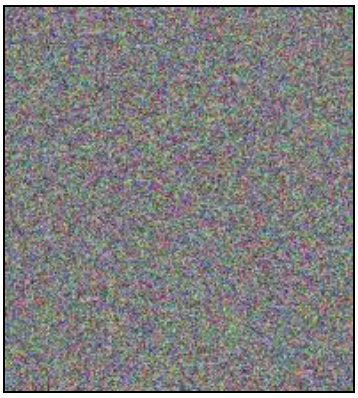

Figure 51

In practice, ECB mode, and deterministic use of a block cipher in general, can introduce the possibility of replay attacks, when proof of identity is in the form of an encrypted value. An eavesdropper may simply replay the encrypted value rather than decode any plaintext.(Kuo-Tsang Huang Chiu Shen 2013)

10.4 : Cipher Block Chaining Mode (CBC)

CBC uses block chaining to provide probabilistic encryption when the key and message are constant.(Dworkin 2001) In this way, the randomness supplied to the encryption function by the IV is chained throughout the execution.(Dworkin 2001) CBC mode achieves this by requiring a unique, random IV to be XORed with the first plaintext before cipher operations.(Paar & Pelzi 2009)

$$E(p_0 \oplus IV,\ k) = c_0 .$$

A produced ciphertext is XORed with the subsequent plaintext before input to the cipher operation.(Paar & Pelzi 2009)

$$E(\ p_i \oplus c_{i-1},\ k) = c_i .$$

Execution

- XOR the IV with the first plaintext block p_0
- Encrypt this value $p_0 \oplus IV$ with key k, to produce ciphertext block c_0
- XOR ciphertext block c_0, with the subsequent plaintext block p_1
- Encrypt this value $p_1 \oplus c_0$ with key k, to produce ciphertext block c_1
- XOR a subsequent plaintext block p_i, with the most recent ciphertext block c_{i-1}
- Encrypt their value $p_i \oplus c_{i-1}$ with key k to produce remaining ciphertext blocks c_i

As decryption is the inverse operation, the current ciphertext block c_i is decrypted then the necessary value is XORed to produce plaintext block p_i.(Paar & Pelzi 2009)

$$D(\ c_i,\ k) \oplus c_{i-1} = p_i$$

For the initial block c_0, the IV is used as the ciphertext block c_{i-1}, to produce plaintext p_0.(Paar & Pelzi 2009)

$$D(\ c_0,\ k) \oplus IV = p_0$$

Cipher Block Chaining (CBC) mode of operation was patented in 1976 by Ehrsam, Meyer, Smith and Tuchman, US Patent 4074066. The IV is a valuable improvement, as even a single bit change causes operation of a general block cipher to be non-deterministic. To make ciphertext, produced with constant plaintext and key, non-deterministic, a unique, random IV must be provided for each execution.(Housley 2004) Under CBC mode, each block of plaintext is XORed with

the previously generated ciphertext block before it is given as input to the cipher. Because the cipher's operation on one block is influenced by its operation on another block, CBC mode is said to display chaining dependency.(Paar & Pelzi 2009)

CBC plaintext encryption is dependant on all previously processed plaintext blocks, as such:(Housley 2004)

- Encryption must be performed serially as each operation provides a necessary unknown input.
- Any transmission error will be propagated throughout, and corrupt following operations.

CBC ciphertext decryption is dependant on the value of the previous ciphertext block, as such:(Housley 2004)

- Decryption is parallelizable as each operation relies on known input values
- Use of an incorrect IV corruptions the first plaintext block but all others are unaffected
- Any transmission error will corrupt only the containing blocks decryption

10.5 : Counter Mode (CTR)

Counter mode uses a block cipher to generate a unique keystream k composed of n size c blocks, where c is the cipher block size, such that $cn = m$.[(Dworkin 2001)] This mode implements encryption to transform a set of input "counters" into a keystream that is then XORed with plaintext units of equal location, resulting in the ciphertext unit at that location.[(Dworkin 2001)] To ensure probabilistic encryption, CTR often takes a random IV to be combined with the counter.[(Ferdinand 2017)]

Execution

- $E(v_0, k) = s_0$ Encrypt counter value v_0, under key k, to produce keystream block s_0
- $p_0 \oplus s_0 = c_0$ XOR plaintext block p_0 and keystream block s_0, producing cipherblock c_0
- $v_0 \rightarrow v_1$ Update the counter value v_1
- $E(v_i, k) = s_i$ Encrypt subsequent values v_i, by key k, to produce keystream block s_i
- $p_i \oplus s_i = c_i$ XOR plaintext block p_i and keystream block s_i, producing cipherblock c_i

Where i is the block index $0 \leq i < q$, and q is the number of size n plaintext blocks such that $qn = m$.[(Paar & Pelzi 2009)]

Counter (CTR) mode of operation was developed by Whitfield Diffie and Martin Hellman and introduced by the IEEE in 1979.[(Diffie Hellman 1979)] To generate the pseudo random keystream, the mode uses successive counter values in place of a random IV.[(Housley 2004)] The counter function used for CTR may be derived a number of ways. Any function is valid so long as it produces a sequence of sufficient length such that a given key is never used with the same counter values.[(Housley 2004)] Common is a simple increment by one counter XORed with a unique IV.[(Housley 2004)] Use of a deterministic input function is only a valid concern if the underlying cipher is weak, it is not the responsibility of a mode of operation to try to compensate.[(Lipmaa Rogaway Wagner 2000)]

Two cases exist with a deterministic input function, like increment by one, to resist chosen-plaintext attacks by which an adversary controls the IV–counter pair to cause a collision.[(Ferdinand 2017)] A random IV may be combined with each sequence value by an invertible operation to produce the counter value, this maintains properties of randomness.[(Ferdinand 2017)] A non-random IV may be used if concatenated with the sequence value by placing the former in the first half and the latter in the second.[(Ferdinand 2017)] A 128 bit example concatenates a 64-bit sequence value to a 64-bit IV to produce each 128-bit counter block.[(Ferdinand 2017)] While many

other methods exist, it is ultimately the user's responsibility to ensure that it is impossible, or highly improbable, that a counter value is ever reused with the same key.(Lipmaa Rogaway Wagner 2000)

One interesting benefit is that CTR encryption is its own inverse operation.(Paar & Pelzi 2009) The decryption process produces an identical decryption key-stream to be utilized in the same way as encryption.(Paar & Pelzi 2009) CTR does not display the chaining dependency of CBC, allowing non-sequential transformation or "random access" for both encryption and decryption.(Housley 2004) This is because each block uses the same key, IV, and initial counter value such that the nth block may be processed by taking the nth offset of the initial counter value. As there is no dependency between cipher instances, each transformation may be executed in parallel.(Housley 2004) CTR mode does not propagate error of transmission, error in a single bit will affect only that bit after transformation.(Housley 2004) However, if the counter offset becomes invalid, so to will the key stream and resulting cipher transformations.(Housley 2004)

Conclusions

I have learned a great deal about the subjects that I intended. Having performed research to identify the properties associated with a cryptographically secure block cipher implementation, progress would entail work toward a complete understanding of the supporting mathematical theory and their broader implications. Ultimately, a further analysis to achieve a functional knowledge of each of the cryptographic primitives presented by the taxonomy in Figure 5 would be ideal.

The scope of this report changed over the course of its execution, becoming increasingly vast. As I was unable to manage the work that I had set for myself, at the suggestion of my Senior Project advisor, I placed work that I could not fully integrate into a future work folder for pursuit after this project.

Delayed work:

Classical Ciphers

- Transposition Ciphers
- Substitution Ciphers
  - Classical Modular Ciphers: Ceaser and Affine Ciphers
  - Polygraphic Substitution: MixColumns and Hill Ciphers

15 Page Timeline of Communication and Cryptographic Application

Cryptologic History

- Advent of Cryptology
- Advent of Cryptanalysis and Frequency Analysis
- Pre Modern
  - WWI: UK Room 40, US Black Chamber
  - WWII: Enigma Machine, PURPLE, UK Bletchley Park, US Sig Int, Polish Cipher Bureau

- The DES Selection Process (Submission Overviews, Design Criteria, Evaluation)

- Public Key Cryptography
- Diffie–Hellman Key Exchange
- Pseudorandom bit generation
- Digital Signatures
- The RSA Cryptosystem
- Message authentication codes
- Cryptographic Hash Functions
- Identification and Entity Authentication
- The AES Selection Process (Submission Overviews, Design Criteria, Evaluation)
- Elliptic-curve Diffie–Hellman key exchange
- Post-quantum cryptography

Public Key Mathematics
- Modular Exponentiation
- Extended Euclidean Algorithm
- Discrete Logarithm Problem
- Elliptic Curve Cryptography

Asymmetric Key Establishment
- Public Key Infrastructure

Efficient GPU Implementation
- Concurrency
- Data Locality
- T-Tables
- BitSlicing

Work Cited

(Greenlaw & Hoover 1998)

Greenlaw, Raymond and Hoover, H.James. 1998. *Fundamentals of the Theory of Computation: Principles and Practice.* Massachusetts: Morgan Kaufman.

(Shannon 1948)

Shannon, C.E. 1948. “A Mathematical Theory of Communication”. Reprinted with corrections from The Bell System Technical Journal, Vol. 27, (July, October). 379–423, 623–656. http://www.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf

(Rowlett 2018)

Rowlett, Russ. 2018. “How Many? A Dictionary of Units of Measurements.” University of North Carolina at Chapel Hill. http://www.ibiblio.org/units/ [shannon /dictS.html#shannon] [bit /dictB.html#bit]

(Buchholz 2000)

Buchholz, Werner.. “IEEE Annals of the History of Computing: Comments, Queries and Debates”. April–June 2000, 69-71. https://web.archive.org/web/20030605004419/http://computer.org/annals/an2000/pdf/a2069.pdf

(Kuphaldt 2001)

Kuphaldt, Tony R. 2001. “All About Circuits: Lessons in Electric Circuits”. EETechMedia. https://www.allaboutcircuits.com/textbook/

1) direct-current/chpt-9/analog-and-digital-signals/ par. 2
2) digital/chpt-1/decimal-versus-binary-numeration/ par. 4
3) digital/chpt-14/introduction-to-digital-communication/

(AAKCT 2018)

Anagnostopoulos, Nikolaos Athansios, Stefan Katzenbeisser, Stephen, Chandy, John and Tehranipoor, Fatemeh. 2018. “An Overview of DRAM-Based Security Primitives”. Cryptography. https://www.mdpi.com/2410-387X/2/2/7/pdf

(BDA 2010)

Blu-ray Disc Association, 2010. "White Paper Blu-ray Disc™ Format 1.C Physical Format Specifications for BD-ROM . http://www.blu-raydisc.com/Assets/Downloadablefile/BD-ROMwhitepaper20070308-15270.pdf

(Woodland & Bernard 1949)

Woodland, Norman and Bernard, Silver. 1949. (US2612994). “Classifying Apparatus and Method”. US Patent Office. https://patents.google.com/patent/US2612994

(Apt 2019)
Apt, Adam Jared. 2019. “Thomas Harriet: English Mathematician and Astronomer”. Encyclopaedia of Britannica. https://www.britannica.com/biography/Thomas-Harriot

(O’Connor & Robertson 2019)
O’Connor, John J. and Robertson, Edmund. 2019. “Harriot and Binary Numbers”. University of St. Andrews Scotland. https://www-history.mcs.st-and.ac.uk/Extras/Harriot_binary_numbers.html

(Shirly 1951)
Shirley, John W. 1951. “Binary Numeration Before Leibniz”, American Journal of Physics, vol. 19, issue 8, 452–454. https://doi.org/10.1119/1.1933042

(Ochulor 2011)
Ochulor, Chinenye Leo. 2011. “Francis Bacon’s Qualification as a Principal Empiricist Philosopher”, Canadian Social Sciences, Vol. 7, No. 5, 229-235. DOI:10.3968/J.css.1923669720110705.270

(Gallup 2010)
Gallup, Elizabeth. 2010. *The Biliteral Cypher of Francis Bacon*, Montana: Kessinger Publishing.

(O’Connor & Robertson 2010)
O’Connor, John J. and Robertson, Edmund. 2010. “Juan Caramuel Y Lobkowitz”. University of St. Andrews Scotland. https://www-history.mcs.st-andrews.ac.uk/Biographies/Caramuel.html

(Smith 2008)
Smith, Justin. 2008. *Leibniz: What Kind of Rationalist.* Heidelberg, Germany. Springer Science + Business Media.

(Wilhelm & Baynes 1967)
Wilhelm, Richard and Baynes, Cary. 1967. *The I Ching or Book of Changes.* New York: Bollingen Foundation Inc.

(ALLM 2018)
Ares, J., Lara, J. Lizcano, D. and Martinez, M.A. 2018. “Who Discovered the Binary System in Arithmetic? Did Leibniz Plagerize Caramuel?” Maryland: National Center of Biotechnical Information. U.S. National Library of Medicine. https://www.ncbi.nlm.nih.gov/pubmed/28281152

(Lande 2014)
Lande, Daniel. 2014. “Development of the Binary Number System and the Foundation of Computer Science”. The Mathematics Enthusiast. Vol. 11, No. 3. Article 6 12-2014. https://scholarworks.umt.edu/cgi/viewcontent.cgi?article=1315&context=tme.

(Strickland 2007)
Strickland, Lloyd. 2007. “Explanation of Binary Arithmetic”. Leibniz’s Translations.com. Die mathematische schriften von Gottfried Wilhelm Leibniz, Vol. 7. Gerhardt, C.I. 223-227. http://www.leibniz-translations.com/binary.htm

(Agarwal & Sen 2014)
Agarwal, Ravi and Sen, Syamal. 2014. *Creators of Mathematical and Computational Science.* New York: Springer Cham.

(Heudin 2008)
Heudin, Jean Claude. 2008. *Les Creatures Artificielles: Des Automates aux Mondes Virtuels.* Paris, France: Odile Jacob.

(DMS 2017)
Datta, Mallika, Mitra, Suman and Sumayun, SK. 2017. “Revival and Digitization of Bygone Woven Design through Textile CAD/CAM System- A Case Study”. West Bengal, India. American International Journal of Research in Science, Technology, Engineering & Mathematics. http://iasir.net/AIJRSTEMpapers/AIJRSTEM17-105.pdf

(Essinger 2004)
Essinger, James. 2004. *Jacquard's Web: How a Hand-Loom Led to the Birth of the Information Age.* Oxford, England: Oxford Press.

(Norman 2019)
Norman. Jeremy. 2019. “A Prayerbook Entirely Woven by the Jacquard Loom: The First Book Produced by a Program or Digitally Produced Book?” History of Information.com. http://www.historyofinformation.com/detail.php?entryid=1870

(Gross. 2015)
Gross, Benjamin. 2015. “The French Connection”. Distillations. Science History Institute. https://www.sciencehistory.org/distillations/the-french-connection

(Halacy 1970)
Halacy, Daniel Stephen. 1970. *Charles Babbage, Father of the Computer.* New York: Crowell-Collier Press.

(Copeland 2017)
Copeland, B. Jack. 2017, "The Modern History of Computing", *The Stanford Encyclopedia of Philosophy* (Winter 2017 Edition), Edward N. Zalta (ed.), https://plato.stanford.edu/archives/win2017/entries/computing-history/

(Swad 2019)
Swade, Doran. “The Babbage Engine”. Computer History Museum. https://www.computerhistory.org/babbage/engines/

(CPRR.org 2014)
CPRR.org 2014, "DID YOU KNOW THAT PUNCHED RAILROAD TICKETS WERE THE FORERUNNERS OF COMPUTERS?" Central Pacific Railroad Photographic History Museum http://cprr.org/Museum/Books/Patton_Made_in_USA.html

(Da Cruz 2001)
Da Cruz, Frank. 2001. “Herman Hollerith”. Columbia University Computing History. http://www.columbia.edu/cu/computinghistory/hollerith.html

(Da Cruz 2019)
Da Cruz, Frank. 2019. “Hollerith 1890 Census Tabulator”. Columbia University Computing History. http://www.columbia.edu/cu/computinghistory/census-tabulator.html

(Satyasikha 2014)
Satyasikha. 2014. “Herman Hollerith”. Engineering and Technology History Wiki. https://ethw.org/Herman_Hollerith

(Rosenbaum 1998)
Rosenbaum, David. 1998. *Market Dominance: How Firms Gain, Hold, or Lose It and the Impact on Economic Performance.* Connecticut: Praeger Publishers.

(ProCon 2013)
ProCon.org. (2013, February 6). Voting Systems & Use: 1980-2012. Retrieved from http://votingmachines.procon.org/view.resource.php?resourceID=000274

(Buchholz 1956)
Buchholz, Werner. (1956-06-11). The Link System (PDF). IBM. pp. 5–6. the original. http://archive.computerhistory.org/resources/text/IBM/Stretch/pdfs/06-07/102632284.pdf

(Buchholz 1962)
Buchholz, Werner. 1962. *Planning a Computer System: Project Stretch.* New York: McGraw Hill Book Company Publishing.

(Swad 2019)
Swade, Doran. 2019 "Internet History of 1960s" Computer History Museum https://www.computerhistory.org/internethistory/1960s/

(Bemer 2000)
Bemer, Bob 2000 Aug 08 "WHY IS A BYTE 8 BITS? OR IS IT?" Computer History Vignettes https://web.archive.org/web/20010411054143/http://www.bobbemer.com/BYTE.HTM

(Koblentz 2004)
Koblentz, Evan Dec. 7, 2004 "LED calculators rule her house" Archive: Computer Collector Newsletter / Technology Rewind, Jan. 2004 - March 2006 http://www.snarc.net/tr/katie-led.htm

(Waterman & Asanovi´c 2017)
Waterman, Andrew Asanovi´c, Krste 2017 "The RISC-V Instruction Set Manual Volume I: User-Level ISA" Document Version 2.2, SiFive Inc., CS Division, EECS Department, University of California, Berkeley
https://content.riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf

(NIST 2019)
NIST Employees. 2019.“SI Units”. National Institute of Standards and Technology – Physical Measurement Laboratory. Maryland.
https://www.nist.gov/pml/weights-and-measures/metric-si/si-units

(McCullagh 2007)
McCullagh, Declan December 5, 2007 "Gigabytes vs. gibibytes class action suit nears end" CBS INTERACTIVE INC.
https://www.cnet.com/news/gigabytes-vs-gibibytes-class-action-suit-nears-end/

(IEC-W 2019)
IEC Employees. 2019. "About the IEC Vision & mission - Welcome to the IEC" International Electrotechnical Commission. https://www.iec.ch/about/

(IEC-G 2019)
IEC Employees. 2019. "About the IEC Vision & mission - Global Reach”. International Electrotechnical Commission. iec.ch/about/globalreach/

(IUCr 1997)
IUCr Employees. 1997. “IUCr 1996 Report – IUPAC Interdivisional Committee on Nomenclature and Symbols (IDCNS)”. International Union of Crystallography.
https://web.archive.org/web/20130613121942/http://www.chester.iucr.org/iucr-top/cexec/rep96/idcns.htm

(IEC 2005)
IEC Employees. September, 2005 "Review of Content Standard Letter symbols to be used in electrical technology IEC 60027" International Electrotechnical Commission.
https://neo.dmcs.pl/ak/IEC_60027-SIUnits.pdf

(Abrahams 2000)
S. C. ABRAHAMS, IUCr Representative *November 2000* "Report of the Executive Committee for 1999 - 15.1. IUPAC Interdivisional Committee on Nomenclature and Symbols (IDCNS)" *Acta Crystallographica Section A: Foundations and Advances, international union of crystallography*
ISSN: 2053-2733 Volume 56 Part 6 Pages 609-642
https://doi.org/10.1107/S0108767300012873

(Buck 2005)
Buck, Jonathan. 2005. “Here Comes Zebi and Yobi”. International Electrotechnical Commission. 2005-08-15.

https://web.archive.org/web/20090912150947/http://www.iec.ch/news_centre/release/nr2005/nr2005.htm

(TMNT 1998)
Barry N. Taylor, Peter J. Mohr, David B. Newell, and E. Tiesinga "The NIST Reference on Constants, Units, and Uncertainty" Fundamental Constants Data Center of the NIST Physical Measurement Laboratory. Online: February 1998 - Last update: June 2019 https://physics.nist.gov/cuu/Units/binary.html

(Thompson & Taylor 2008)
Thompson, Ambler and Taylor, Barry N. March 2008 "Guide for the Use of the International System of Units (SI)" NIST Special Publication 811 2008 Edition

(IEEE 2009)
IEEE Employees. 2009. "1541-2002 - IEEE Standard for Prefixes for Binary Multiples". International Electrical and Electronics Engineering. 18 Sept. 2009
Electronic ISBN: 978-0-7381-3386-7 DOI: 10.1109/IEEESTD.2009.5254933

(Leontiou 2011)
Leontiou, Andrea 2/10/2011 "World's shift from analog to digital is nearly complete" TechNewsDaily NBCNews.com
http://www.nbcnews.com/id/41516959/ns/technology_and_science-innovation/t/worlds-shift-analog-digital-nearly-complete/

(Samson 1999)
Samson December 1999 "Digital Signals - Fundamentals"
https://www.samson.de/document/l150en.pdf

(Stallings 2007)
Stallings, William 2007 "Chapter 4 Transmission Media" Data and Computer Communications Eighth Edition
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.463.7540&rep=rep1&type=pdf

(Van Tilborg & Jajodia, 2011)
Van Tilborg, Heck and Jajodia, Sushil (Editors). 2011. *Encyclopedia of Cryptography and Security. (Change to Esc).* Netherlands: Springer Science + Business Media.

(Menezes Van Oorshot & Vanstone 1997)
Menezes, Alfred J., Van Oorshot, Paul C., and Vanstone, Scott. 1997. *Handbook of Applied Cryptography (Discrete Mathematics and Its Applications).* Florida: CRC Press.

(Paar & Pelzi 2009)
Paar, Christopher and Pelzi, Jan. 2009. *Understanding Cryptography: A Textbook for Students and Practitioners.* Heidelberg, Germany: Springer Technology.

(Stross 2006)
Stross, Randall 2006 "Theater of the Absurd at the T.S.A." The New York Times Company https://www.nytimes.com/2006/12/17/business/yourmoney/17digi.html

(Scarfone Jansen Tracy 2008)
Scarfone, Karen Jansen, Wayne Tracy, Miles 2008 "Guide to General Server Security NIST Special Publication 800-123 Computer Security Division Information Technology Laboratory National Institute of Standards and Technology https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-123.pdf

(Raymond 2000)
Raymond, Eric Steven 2000 "The Cathedral and the Bazaar - Release Early, Release Often" http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/ar01s04.html

(Germain 2016)
Germain, Jack M. Aug 25, 2016 "25 Years of Linux: What a Long, Strange Trip It's Been" Linux Insider https://www.linuxinsider.com/story/83838.html

(Gergersen 2017)
Gergersen Erik 2017 Data Encryption Encyclopaedia of Britannica. https://www.britannica.com/technology/data-encryption

(Fabien Petitcolas 1997)
Petitcolas, Fabien 1997 "Kerckhoffs' principles from « La cryptographie militaire »" The information hiding homepage https://www.petitcolas.net/kerckhoffs/index.html

(Mann 2002)
Mann, Charles. 2002. “Technology: Homeland Insecurity”. The Atlantic Monthly. (September edition).

(Ahmed Al-Vahed 2011)
Ahmed Al-Vahed, Haddad Sahhavi 2011 "An overview of modern cryptography" Semantic Scholar DOI:10.1007/0-387-26090-0_3

(Lafourcade 2013)
Pascal Lafourcade. 2013 Computer Aided Security for Cryptographic Primitives, Voting protocols, and Wireless Sensor Networks. Cryptography and Security [cs.CR]. Université de Grenoble, 2012. fftel00807568f

(Kotzanikolaou & Douligeris 2006)
Kotzanikolaou, Panayiotis and Douligeris, Christos. 2006. “Cryptography Primer: Introduction to Cryptographic Principles and Algorithms”. 460-479. https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470099742.app1

(Bellovin 2011)
Bellovin Steven M. 2011 "Frank Miller: Inventor of the One-Time Pad" Cryptologia Volume 35, Issue 3 p. 203-222 https://doi.org/10.1080/01611194.2011.583711

(Vernam 1919)
Vernam G. S. 1919 "Secret Signaling System" 1,310,719 Patented July 22, 1919. https://patents.google.com/patent/US1310719

(Klein 2003)
Klein, Melville. 2003. “Securing Record Communications. The TSEC/KW-26.” National Security Administration. Retrieved 2012-04-12. http://www.jproc.ca/crypto/kw26.pdf

(Kahn 1996)
Kahn, David. 1996. *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet.* New York: Scribner Press. 744.

(GBCGMV 2002)
Golomb, Berlekamp, Cover, Gallager, Massey, and Viterbi 2002. "Claude Elwood Shannon" Notices of the AMS VOLUME 49, NUMBER 1 http://www.ams.org/notices/200201/fea-shannon.pdf

(Shannon 1949)
Shannon C. E. 1949 "Communication Theory of Secrecy Systems" Bell System Technical Journal http://netlab.cs.ucla.edu/wiki/files/shannon1949.pdf

(Holden 2017)
Holden, Joshua. 2017. *The Mathematics of Secrets: Cryptography from Caesar Ciphers to Digital Encryption.* New Jersey: Princeton University Press.

(Reuvers & Simons 2013)
Reuvers, Paul & Simons, Marc 2013. "One-Time Pad (OTP)" Last changed: Tuesday, 29 January 2013 cryptomuseum.com https://web.archive.org/web/20140314175211/http://www.cryptomuseum.com/crypto/otp.htm

(Smith 2007)
Smith Rick 2007 "ONE-TIME PADS" JUNE 9, 2007 CRYPTOSMITH https://cryptosmith.com/2007/06/09/one-time-pads/

(Hannan & Asif 2017)
Shaikh Abdul Hannan and Ali Mir Arif Mir Asif 2017 Analysis of Polyalphabetic Transposition Cipher Techniques used for Encryption and Decryption International Journal of Computer Science and Software Engineering (IJCSSE), Volume 6, Issue 2, February 2017 ISSN (Online): 2409-4285 www.IJCSSE.org Page: 41-46 http://ijcsse.org/published/volume6/issue2/p3-V6I2.pdf

(Houtven 2013)
Houtven, Laurens Van 2013. Crypto 101 Copyright 2013-2017 https://www.crypto101.io/Crypto101.pdf

(Branstad 1978)
Branstad, Dennis K.1978. "Computer Security and the Date Encryption Standard" NBS Special Publication 500-27 U.S. DEPARTMENT OF COMMERCE National Bureau of Standards https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nbsspecialpublication500-27.pdf

(Burr 1977)
Burr, William. 1977. “Data Encryption Standard”. 250-253. https://nvlpubs.nist.gov/nistpubs/sp958-lide/250-253.pdf

(Keliher 1997)
Keliher Liam 1997. Substitution-Permutation Network Cryptosystems Using Key-Dependent S-Boxes Queen's University

(Simmons 2009)
Simmons J., Gustavus. 2009 “Data Encryption Standard: Cryptology”. Encyclopaedia Britannica. https://www.britannica.com/topic/Data-Encryption-Standard

(Stallings 2017)
Stallings, William. 2017. “Cryptography and Network Security Principals and Practices”. Seventh Edition Global Edition Pearson Education Limited 2017 ISBN 10:1-292-15858-1

(MHFP 2015)
Charalampos Manifavas, George Hatzivasilis, Konstantinos Fysarakis and Yannis Papaefstathiou A survey of lightweight stream ciphers for embedded systems Security Comm. Networks 2016; 9:1226–1246 21 December 2015 https://onlinelibrary.wiley.com/ DOI: 10.1002/sec.1399

(Britannica 2016)
Editors of Encyclopaedia of Britannica. 2016 “Cipher: Cryptology”. Encyclopaedia Britannica. https://www.britannica.com/topic/cipher

(Asif Buchanan Li 2018)
Asif, Rameez Buchanan, William J. Li, Shancang 2018 Lightweight cryptography methods The Cyber Academy, Edinburgh Napier University, UK University of the West of England, UK DOI: 10.1080/23742917.2017.1384917

(Brown & Seberry 1990)
Brown, Lawrence Seberry, Jennifer 1990 ON THE DESIGN OF PERMUTATION P IN DES TYPE CRYPTOSYSTEMS Department of Computer Science University College, UNSW, Australian Defence Force Academy https://link.springer.com/content/pdf/10.1007%2F3-540-46885-4_71.pdf

(Kopal 2018)
Kopal Nils 2018 Solving Classical Ciphers with CrypTool 2 Applied Information Security University of Kassel http://www.ep.liu.se/ecp/149/010/ecp18149010.pdf

(GJMN 2015)
Jian Guo, Jeremy Jean, Nicky Mouha, and Ivica Nikoli´c 2015 More Rounds, Less Security? 1 Nanyang Technological University, Singapore https://eprint.iacr.org/2015/484.pdf

(Dulaney & Easttom 2017)
Dulaney, Emmett and Easttom, Chuck. 2017. *CompTIA Security+ Study Guide: Exam SY0-501.* California: Sybex Inc.7th Ed.

(PSP, 2016)
Pritam Singh Patel, Ravendra Ratan Singh , Satya Patra 2016 Secure Communications over Insecure Networks Using Cryptogram ISAR - International Journal of Electronics and Communication Ethics - Volume 1 Issue 1, Jan – Feb 2016

(LFS 2018)
Leech, David P., Ferris, Stacey and Scott, John T. 2018. "The Economic Impacts of the Advanced Encryption Standard, 1996 - 2017". National Institute of Standards and Technology. https://nvlpubs.nist.gov/nistpubs/gcr/2018/NIST.GCR.18-017.pdf

(NIST 2018)
NIST 2018. NIST's Encryption Standard Has Minimum $250 Billion Economic Benefit, According to New Study NIST September 19, 2018 https://www.nist.gov/news-events/news/2018/09/nists-encryption-standard-has-minimum-250-billion-economic-benefit

(Gilmore 2005)
Gilmore John 2005 DES (Data Encryption Standard) Review at Stanford University September 20, 2005 (updated August 20, 2009, March 20, 2012, and December 21, 2015) http://www.toad.com/des-stanford-meeting.html

(Almunawar 2001)
Almunawar Mohammad Nabil SECURING ELECTRONIC TRANSACTIONS TO SUPPORT E-COMMERCE Faculty of Business, Economics & Policy Studies Universiti Brunei Darussalam https://arxiv.org/ftp/arxiv/papers/1207/1207.4292.pdf

(Chernev 2019)
Chernev, Bobby. 2019. "What Is AES and Why You Already Love It". TechJury. https://techjury.net/blog/what-is-aes/

(Buchmann 2001)
Buchmann, Johannes. 2001. *Introduction to Cryptography (Undergraduate Texts in Mathematics).* New York: Springer Publishing. 1st edition.

(Daemen & Rijmen 2002)
Daemen, Joan and Rijmen, Vincent. 2002. *The Design of RijndaeL: AES - The Advanced Encryption Standard (Information Security and Cryptography).* New York: Springer Publishing.

(NIST 2001)
NIST 2001. "Announcing the Advanced Encryption Standard (AES)". Federal Information Processing Standards Publication 197. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

(Gladman 2003)
Gladman, Brian. 2003. "A Specification for Rijndael, the AES". v3.11, 12th. http://asmaes.sourceforge.net/rijndael/rijndaelImplementation.pdf

(Farwa Shah Idrees 2016)
Farwa, Shabieh, Shah, Tariq and Idrees, Lubna. 2016. "A Highly Nonlinear S-box Based on a Fractional Linear Transformation". Farwa et al. SpringerPlus (2016) 5:1658 DOI 10.1186/s40064-016-3298-7

(Grocholewska-Czurylo 2011)
Grocholewska-Czurylo, Anna. 2011. "Cryptographic properties of modified AES-like S-boxes". Lublin, Poland. Institute of Control and Information Engineering, Pozna´n University of Technology. Annals UMCS. 37-48. https://pdfs.semanticscholar.org/e260/29fbad216db2428eaeb9aca7f074973f0ec1.pdf

(Nyberg 1991)
Nyberg, Kaisa. 1991. "Perfect nonlinear S-boxes". Finnish Defense Force and University of Helsinki. https://link.springer.com/content/pdf/10.1007%2F3-540-46416-6_32.pdf

(Shannon 1945)
Shannon, Claude. 1945. "A Mathematical Theory of Cryptography". Bell System Technical Memo MM 45-110-02, Sept. 1. https://www.iacr.org/museum/shannon/shannon45.pdf

(PM 2015)
Prime.mover 2015. "Definition:Set" Proof Wiki Last Modified 09/03/2019 https://proofwiki.org/wiki/Definition:Set

(SKABMB 2010)
Sakalli, M. Togla, Karaahmetoglu, Osman, Aslan, Bora, Bulus, Erkan, Mesut, Andak, and Buyusaracoglu, Fatima . 2010. "On the Algebraic Expression of the AES S-Box Like S-Boxes". Turkey. Communications in Computer and Information Science. https://www.academia.edu/19213177/On_the_Algebraic_Expression_of_the_AES_S-Box_Like_S-Boxes

(Easttom 2014.)
Easttom, Chuck. 2014. "A Guideline for Designing Cryptographic S-Boxes". https://pdfs.semanticscholar.org/7ae7/bcad617a7106afabc0ee7f29b16b6cadcb22.pdf

(Katiyar Jeyanthi 2019)
Katiyar, Shishir and Jeyanthi, N. 2019. “Pure Dynamic S-Box Construction”. India: VIT University. International Journal for Computers. https://www.iaras.org/iaras/filedownloads/ijc/2016/006-0005.pdf

(Krishnamurthy Ramaswamy 2008)
Krishnamurthy, G.N. and Ramaswamy, V. 2008. “Making AES Stronger: AES with Key Dependent S-Box”. IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.9, ed. September. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.572.140&rep=rep1&type=pdf

(Cid Murphy Robshaw 2006)
Cid, Carlos, Murphy, Sean and Robshaw, Matthew. 2006. *Algebraic Aspects of the Advanced Encryption Standard.* New York: Springer Science + Business Media LLC.

(Gallian 2016)
Gallian, Joseph. 2016. *Contemporary Abstract Algebra.* California: Cengage Learning. 9th edition.

(Dong 2010)
Dong, Changyu. 2010 “Math in Network Security: A Crash Course”. https://www.doc.ic.ac.uk/~mrh/330tutor/index.html

(Diffie Hellman 1979)
Diffie, Whitfield and Hellman, Martin. 1979. “Privacy and Authentication: An Introduction to Cryptography”. Proceedings of the IEEE, Vol. 67, No. 3, March https://ee.stanford.edu/~hellman/publications/32.pdf

(Lipmaa Rogaway Wagner 2000)
Lipmaa, Helger, Rogaway, Phillip and Wagner, David. 2000. “Comments to NIST concerning AES Modes of Operations: CTR-Mode Encryption”. Berkeley: University of California. https://web.cs.ucdavis.edu/~rogaway/papers/ctr.pdf

(Ferdinand 2017)
Ferdinand Sibleyras. 2017. Cryptanalysis of the Counter mode of operation. Cryptography and Security [cs.CR]. HAL id: hal-01662040f https://hal.inria.fr/hal-01662040/document

(Schiller Crocker 2005)
Schiller, J. Crocker, S. 2005. Randomness Requirements for Security Network Working Group Request for Comments: 4086 https://tools.ietf.org/html/rfc4086

(Housley 2009)
Housley, R. 2009. "Cryptographic Message Syntax (CMS)" Network Working Group.Request for Comments: 5652 https://tools.ietf.org/html/rfc5652

(Housley 2004)
Housley, R. 2004. “Using Advanced Encryption Standards (AES) Countermode with IPsec Encapsulating Security Payload (ESP)”. Network Working Group. Request for Comments: 3686 https://tools.ietf.org/html/rfc3686#section-2.1

(Ferguson Schneier 2003)
Ferguson, Neils and Schneier, Bruce. 2003. *Practical Cryptography.* Indiana: Wiley Publishing Inc.

(Dworkin 2001)
Dworkin, Morris. 2001. “Recommendations for Block Cipher Modes of Operation: Methods and Techniques”. Maryland: National Institute of Standards and Technology. NIST Special Publication 800-38A. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf

(Kuo-Tsang Huang Chiu Shen 2013)
Kuo-Tsang Huang, Kuo-Tsang, Chiu, Jung-Hui and Shen, Sung-Shiou. 2013. “A Novel Structure with Dynamic Operation Mode For Symmetric Key-Block Ciphers”. International Journal of Network Security & Its Applications (IJNSA), Vol.5, No.1, January. http://airccse.org/journal/nsa/0113nsa02.pdf

(Jackson 2017)
Jackson, Nicholas. 2017. “A Course in Abstract Algebra”. Warwick, England. http://dl.icdst.org/pdfs/files1/54f189606fea45076e942b7166fa9af1.pdf

(Cohn 1981)
Cohn, P. M. 1981. *Universal Algebra.* New York: Springer Science + Business Media LLC.

(Smart 2016)
Smart, Nigel P. 2016. *Cryptography Made Simple.* New York: Springer Science + Business Media LLC.

(O’Leary 2015)
O’Leary, Michael. 2015. *A First Course in Mathematical Logic and Set Theory.* Indiana: Wiley Publishing Inc. 1st ed.

(Sedgewick Wayne 2011)
Sedgewick, Robert and Wayne, Kevin. 2011. *Algorithms.* New Jersey: Pearson Education Inc. 4th Ed.

(Fedler 2013)
Fedler, Rapahel. 2013. “Padding Oracle Attacks”. Seminar Innovative Internet Technologies and Mobile Communications, SS 2013 Chair for Network Architectures and Services Department of Computer Science, Technische Universität München. https://pdfs.semanticscholar.org/2b88/e2925523e46b523fd98b8f2b349defcf76f7.pdf

(Bello Danjuma Simon 2018)
Bello, M., Danjuma, Mustapha and Simon, Bulus. 2018. “On the Simplicity of Permutation Groups”. Global Scientific Journals. Vol. 6, Issue. 1, January. http://www.globalscientificjournal.com/researchpaper/ON-THE-SIMPLICITY-OF-PERMUTATION-GROUPS.pdf

(Mcanet 2009)
Mcanet 2009. "Ruido señal digital" Own work https://commons.wikimedia.org/wiki/File:Imagen_4.png

(Bilou 2010)
Bilou 2010. "Schematic depiction of the matrix product AB of two matrices A and B." Own work https://commons.wikimedia.org/wiki/File:Matrix_multiplication_diagram_2.svg

(Wiki 2002)
2002 "Matrix multiplication" Wikimedia Foundation, Inc. last edited on 7 September 2019 https://en.wikipedia.org/wiki/Matrix_multiplication#Definition

(Wiki 2001)
Wikipedia 2001 "Specific units of IEC 60027-2 A.2 and ISO/IEC 80000" Wikimedia Foundation, Inc. last edited on 28 August 2019 https://en.wikipedia.org/wiki/Binary_prefix#cite_ref-70