Information Security and Cryptography

Joan Daemen
Vincent Rijmen

The Design of Rijndael

The Advanced Encryption Standard (AES)

Second Edition

Springer

Information Security and Cryptography

More information about this series at http://www.springer.com/series/4752

Joan Daemen • Vincent Rijmen

The Design of Rijndael

The Advanced Encryption Standard (AES)

Second Edition

Springer

Joan Daemen
Digital Security Group
Radboud University Nijmegen
Nijmegen, The Netherlands

Vincent Rijmen
COSIC Group
KU Leuven
Heverlee, Belgium

ISSN 1619-7100 ISSN 2197-845X (electronic)
Information Security and Cryptography
ISBN 978-3-662-60768-8 ISBN 978-3-662-60769-5 (eBook)
https://doi.org/10.1007/978-3-662-60769-5

This Springer imprint is published by the registered company Springer-Verlag GmbH, DE part of Springer Nature.
The registered company address is: Heidelberger Platz 3, 14197 Berlin, Germany

Foreword

Rijndael was the surprise winner of the contest for the new Advanced Encryption Standard (AES) for the United States. This contest was organized and run by the National Institute of Standards and Technology (NIST) beginning in January 1997; Rijndael was announced as the winner in October 2000. It was the "surprise winner" because many observers (and even some participants) expressed scepticism that the US government would adopt as an encryption standard any algorithm that was not designed by US citizens.

Yet NIST ran an open, international, selection process that should serve as a model for other standards organizations. For example, NIST held their 1999 AES meeting in Rome, Italy. The five finalist algorithms were designed by teams from all over the world.

In the end, the elegance, efficiency, security, and principled design of Rijndael won the day for its two Belgian designers, Joan Daemen and Vincent Rijmen, over the competing finalist designs from RSA, IBM, Counterpane Systems, and an English/Israeli/Danish team.

This book is the story of the design of Rijndael, as told by the designers themselves. It outlines the foundations of Rijndael in relation to the previous ciphers the authors have designed. It explains the mathematics needed to understand the operation of Rijndael, and it provides reference C code and test vectors for the cipher.

Most importantly, this book provides justification for the belief that Rijndael is secure against all known attacks. The world has changed greatly since the DES was adopted as the national standard in 1976. Then, arguments about security focused primarily on the length of the key (56 bits). Differential and linear cryptanalysis (our most powerful tools for breaking ciphers) were then unknown to the public. Today, there is a large public literature on block ciphers, and a new algorithm is unlikely to be considered seriously unless it is accompanied by a detailed analysis of the strength of the cipher against at least differential and linear cryptanalysis.

This book introduces the "wide trail" strategy for cipher design, and explains how Rijndael derives strength by applying this strategy. Excellent resistance to differential and linear cryptanalysis follows as a result. High efficiency is also a result, as relatively few rounds are needed to achieve strong security.

The adoption of Rijndael as the AES is a major milestone in the history of cryptography. It is likely that Rijndael will soon become the most widely used cryptosystem in the world. This wonderfully written book by the designers themselves is a "must read" for anyone interested in understanding this development in depth.

Ronald L. Rivest
Viterbi Professor of Computer Science
MIT

Preface

This book is about the design of Rijndael, the block cipher that became the Advanced Encryption Standard (AES). According to the 'Handbook of Applied Cryptography' [110], a block cipher can be described as follows:

> A block cipher is a function which maps n-bit plaintext blocks to n-bit ciphertext blocks; n is called the block length. [...] The function is parameterized by a key.

Although block ciphers are used in many interesting applications, such as e-commerce and e-security, this book is *not* about applications. Instead, this book gives a detailed description of Rijndael and explains the design strategy that was used to develop it.

Structure of this book

When we wrote this book, we had basically two kinds of readers in mind. Perhaps the largest group of readers will consist of people who want to read a full and unambiguous description of Rijndael. For those readers, the most important chapter of this book is Chap. 3, which gives its comprehensive description. In order to follow our description, it might be helpful to read the preliminaries given in Chap. 2. Advanced implementation aspects are discussed in Chap. 4. A short overview of the AES selection process is given in Chap. 1.

A large part of this book is aimed at readers who want to know *why* we designed Rijndael in the way we did. For them, we explain the ideas and principles underlying the design of Rijndael, culminating in our wide trail design strategy. In Chap. 5 we explain our approach to block cipher design and the criteria that played an important role in the design of Rijndael. Our design strategy has grown out of our experiences with linear and differential cryptanalysis, two cryptanalytical attacks that have been applied with some success to the previous standard, the Data Encryption Standard (DES). In Chap. 6 we give a short overview of the DES and of the differential and the linear attacks that are applied to it. Our framework to describe linear cryptanalysis is explained in Chap. 7; differential cryptanalysis is described

in Chap. 8. Finally, in Chap. 9, we explain how the wide trail design strategy follows from these considerations.

Chap. 10 gives an overview of the published attacks on reduced-round variants of Rijndael. Chap. 11 gives an overview of ciphers related to Rijndael. We describe its predecessors and discuss their similarities and differences. This is followed by a short description of a number of block ciphers that have been strongly influenced by Rijndael and its predecessors.

In Chap. 12 we show how linear and differential analysis can be applied to ciphers that are defined in terms of finite field operations rather than Boolean functions. In Chap. 13 we discuss extensions of differential and linear cryptanalysis. In Chap. 14 we study the probability of differentials and trails over two rounds of Rijndael, and in Chap. 15 we define plateau trails. To assist programmers, Appendix A lists some tables that are used in various descriptions of Rijndael, Appendix B gives a set of test vectors, and Appendix C consists of an example implementation of Rijndael in the C programming language.

Acknowledgments

This book would not have been written without the support and help of many people. It is impossible for us to list all people who contributed along the way. Although we probably will make oversights, we would like to name some of our supporters here.

First of all, we would like to thank the many cryptographers who contributed to developing the theory on the design of symmetric ciphers, and from who we learned much of what we know today. We would like to mention explicitly the people who gave us feedback in the early stages of the design process: Johan Borst, Antoon Bosselaers, Paulo Barreto, Craig Clapp, Erik De Win, Lars R. Knudsen and Bart Preneel.

Elaine Barker, James Foti and Miles Smid, and all the other people at NIST who worked very hard to make the AES process possible and visible.

The moral support of our family and friends, without whom we would never have persevered.

Brian Gladman, who provided test vectors.

Othmar Staffelbach, Elisabeth Oswald, Lee McCulloch and other proofreaders, who provided very valuable feedback and corrected numerous errors and oversights.

The financial support of KU Leuven, the Fund for Scientific Research – Flanders (Belgium), Banksys, Proton World and Cryptomathic is also greatly appreciated.

November 2001 *Joan Daemen and Vincent Rijmen*

Preface to the second edition

This edition contains updates of several chapters as well as four new chapters (Chaps. 12 to 15). We adapted our text to new terminology that has come into use since the first edition and removed some material that is now obsolete, including the original Appendix A (Propagation Analysis in Galois Fields) and the original Appendix B (Trail Clustering).

We thank Ronan Nugent of Springer for his persistent encouragements to finalize this second edition. We thank Dave, Eric Bach, Nicolas T. Courtois, Praveen Gauravaram, Jorge Nakahara Jr., Ralph Wernsdorf, Shengbo Xu and Uyama Yasumasa for pointing out errors in the first edition of this book. We thank Bart Mennink for proofreading some of the updates. All remaining errors and those newly introduced are of course our own.

July 2019 *Joan Daemen and Vincent Rijmen*

Contents

1. The Advanced Encryption Standard Process

The main subject of this book would probably have remained an esoteric topic of cryptographic research — with a name unpronounceable to most of the world — without the Advanced Encryption Standard (AES) process. Therefore, we thought it proper to include a short overview of the AES process.

1.1 In the Beginning ...

In January 1997, the US National Institute of Standards and Technology (NIST) announced the start of an initiative to develop a new encryption standard: the AES. The new encryption standard was to become a Federal Information Processing Standard (FIPS), replacing the old Data Encryption Standard (DES) and Triple DES.

Unlike the selection process for the DES, the Secure Hash Algorithm (SHA-1) and the Digital Signature Algorithm (DSA), NIST had announced that the AES selection process would be open. Anyone could submit a candidate cipher. Each submission, provided it met the requirements, would be considered on its merits. NIST would not perform any security or efficiency evaluation itself, but instead invited the cryptology community to mount attacks and try to cryptanalyze the different candidates, and anyone who was interested to evaluate implementation cost. All results could be sent to NIST as public comments for publication on the NIST AES web site or be submitted for presentation at AES conferences. NIST would merely collect contributions, using them as the basis for their selection. NIST would motivate their choices in evaluation reports.

1.2 AES: Scope and Significance

The official scope of a FIPS standard is quite limited: the FIPS only applies to the US Federal Administration. Furthermore, the new AES would only be used for documents that contain *sensitive but not classified* information.

J. Daemen, V. Rijmen, *The Design of Rijndael*, Information Security and Cryptography,
https://doi.org/10.1007/978-3-662-60769-5_1

However, it was anticipated that the impact of the AES would be much larger than this: for the AES is the successor of the DES, the cipher that ever since its adoption has been used as a worldwide de facto cryptographic standard by banks, administrations and industry.

The major factors for the quick acceptance of Rijndael are that it is available royalty-free, and that it can be implemented easily on a wide range of platforms without reducing bandwidth in a significant way.

1.3 Start of the AES Process

In September 1997, the final request for candidate nominations for the AES was published. The minimum functional requirements asked for symmetric block ciphers capable of supporting block lengths of 128 bits and key lengths of 128, 192 and 256 bits. An early draft of the AES functional requirements had asked for block ciphers also supporting block sizes of 192 and 256 bits, but this requirement was dropped later on. Nevertheless, since the request for proposals mentioned that extra functionality in the submissions would be received favorably, some submitters decided to keep the variable block length in the designs. (Examples include RC6 and Rijndael.)

NIST declared that it was looking for a block cipher *as secure as Triple DES, but much more efficient.* Another mandatory requirement was that the submitters agreed to make their cipher available on a worldwide royalty-free basis if it were to be selected as the AES. In order to qualify as an official AES candidate, the designers had to provide:

1. A complete written specification of the block cipher in the form of an algorithm.
2. A reference implementation in ANSI C, and mathematically optimized implementations in ANSI C and Java.
3. Implementations of a series of known-answer and Monte Carlo tests, as well as the expected outputs of these tests for a correct implementation of their block cipher.
4. Statements concerning the estimated computational efficiency in both hardware and software, the expected strength against cryptanalytic attacks, and the advantages and limitations of the cipher in various applications.
5. An analysis of the cipher's strength against known cryptanalytic attacks.

It turned out that the required effort to produce a 'complete and proper' submission package would already filter out several of the proposals. Early in the submission stage, the Cryptix team announced that they would provide Java implementations for all submitted ciphers, as well as Java implementations

of the known-answer and Monte Carlo tests. This generous offer took some weight off the designers' shoulders, but still the effort required to compile a submission package was too heavy for some designers. The fact that the AES Application Programming Interface (API), which all submissions were required to follow, was updated twice during the submission stage increased the workload. Table 1.1 lists (in alphabetical order) the 15 submissions that were completed in time and accepted.

Table 1.1. The 15 AES candidates accepted for the first evaluation round

Submissions	Submitter(s)	Submitter type
CAST-256	Entrust (CA)	Company
Crypton	Future Systems (KR)	Company
DEAL	Outerbridge and Knudsen (USA–DK)	Researchers
DFC	ENS-CNRS (FR)	Researchers
E2	NTT (JP)	Company
FROG	TecApro (CR)	Company
HPC	Schroeppel (USA)	Researcher
LOKI97	Brown et al. (AU)	Researchers
Magenta	Deutsche Telekom (DE)	Company
Mars	IBM (USA)	Company
RC6	RSA (USA)	Company
Rijndael	Daemen and Rijmen (BE)	Researchers
SAFER+	Cylink (USA)	Company
Serpent	Anderson, Biham and Knudsen (UK–IL–DK)	Researchers
Twofish	Counterpane (USA)	Company

1.4 The First Round

The selection process was divided into several stages, with a public workshop to be held near the end of each stage. The process started with a *submission stage*, which ended on 15 May 1998. All accepted candidates were presented at *The First Advanced Encryption Standard Candidate Conference*, held in Ventura, California, on 20-22 August 1998. This was the official start of the first evaluation round, during which the international cryptographic community was asked for comments on the candidates.

1.5 Evaluation Criteria

The evaluation criteria for the first round were divided into three major categories: security, cost and *algorithm and implementation characteristics*. NIST invited the cryptology community to mount attacks and try to cryptanalyze the different candidates, and anyone interested to evaluate implementation cost. The result could be sent to NIST as public comments or be submitted for presentation at the second AES conference. NIST collected all contributions and would use these to select five finalists. In the following sections we discuss the evaluation criteria.

1.5.1 Security

Security was the most important category, but perhaps the most difficult to assess. Only a small number of candidates showed some theoretical design flaws. The large majority of the candidates fell into the category 'no weakness demonstrated'.

1.5.2 Costs

The 'costs' of the candidates were divided into different subcategories. A first category was formed by costs associated with intellectual property (IP) issues. First of all, each submitter was required to make his cipher available for free if it were to be selected as the AES. Secondly, each submitter was also asked to make a signed statement that he would not claim ownership or exercise patents on ideas used in *another submitter's proposal* that would eventually be selected as the AES. A second category of 'costs' was formed by costs associated with the implementation and execution of the candidates. This covers aspects such as computational efficiency, program size and working memory requirements in software implementations, and chip area in dedicated hardware implementations.

1.5.3 Algorithm and Implementation Characteristics

The category *algorithm and implementation characteristics* grouped a number of features that are harder to quantify. The first one is *versatility*, meaning the ability to be implemented efficiently on different platforms. At one end of the spectrum the AES should fit 8-bit micro-controllers and smart cards, which have limited storage for the program and a very restricted amount of RAM for working memory. At the other end of the spectrum the AES should be implementable efficiently in dedicated hardware, e.g. to provide on-the-fly encryption/decryption of communication links at gigabit-per-second rates. In

between there is the whole range of processors that are used in servers, workstations, PCs, palmtops etc., which are all devices in need of cryptographic support. A prominent place in this range is taken by the Pentium family of processors due to its presence in most personal computers.

A second feature is *key agility*. In most block ciphers, key setup takes some processing. In applications where the same key is used to encrypt large amounts of data, this processing is relatively unimportant. In applications where the key often changes, such as the encryption of Internet Protocol (IP) packets in Internet Protocol Security (IPSEC), the overhead due to key setup may become quite relevant. Obviously, in those applications it is an advantage to have a fast key setup.

Finally, there is the criterion of *simplicity*, which may even be harder to evaluate than cryptographic security. Simplicity is related to the size of the description, the number of different operations used in the specification, the symmetry or lack of symmetry in the cipher and the ease with which the algorithm can be understood. All other things being equal, NIST considered it to be an advantage for an AES candidate to be more simple for reasons of ease of implementation and confidence in security.

1.6 Selection of Five Finalists

In March 1999, the second AES conference was held in Rome, Italy. The remarkable fact that a US Government department organized a conference on a future US Standard in Europe is easily explained. NIST chose to combine the conference with the yearly Fast Software Encryption Workshop, which had for the most part the same target audience and was scheduled to be in Rome.

1.6.1 The Second AES Conference

The papers presented at the conference ranged from crypto-attacks, cipher cross-analysis, smart-card-related papers, and so-called *algorithm observations*. In the session on cryptographic attacks, it was shown that FROG, Magenta and LOKI97 did not satisfy the security requirements imposed by NIST. For DEAL it was already known in advance that the security requirements were not satisfied. For HPC weaknesses had been demonstrated in a paper previously sent to NIST. This eliminated five candidates.

Some cipher cross-analysis papers focused on performance evaluation. The paper of B. Gladman [65], a researcher who had no link with any submission, considered performance on the Pentium processor. From this paper it became clear that RC6, Rijndael, Twofish, MARS and Crypton were the five fastest ciphers on this processor. On the other hand, the candidates DEAL, Frog,

Magenta, SAFER+ and Serpent appeared to be problematically slow. Other papers by the Twofish team (Bruce Schneier et al.) [138] and a French team of 12 cryptographers [10] essentially confirmed this.

A paper by E. Biham warned that the security margins of the AES candidates differed greatly and that this should be taken into account in the performance evaluation [21]. The lack of speed of Serpent (with E. Biham in the design team) was seen to be compensated for by a very high margin of security. Discussions on how to measure and take into account security margins lasted until after the third AES conference.

In the session on smart cards there were two papers comparing the performance of AES candidates on typical 8-bit processors and a 32-bit processor: one by G. Keating [78] and one by G. Hachez et al. [69]. From these papers and results from other papers, it became clear that some candidates simply did not fit onto a smart card and that Rijndael was by far the best suited for this platform. In the same session there were some papers that discussed power analysis attacks and the suitability of the different candidates for implementations that can resist against these attacks [27, 39, 49].

Finally, in the *algorithm observations* session, there were a number of papers in which AES submitters re-confirmed their confidence in their submission by means of a considerable amount of formulas, graphs and tables and some *loyal cryptanalysis* (the demonstration of having found no weaknesses after attacks on their own cipher).

1.6.2 The Five Finalists

After the workshop there was a relatively calm period that ended with the announcement of the five candidates by NIST in August 1999. The finalists were (in alphabetical order): MARS, RC6, Rijndael, Serpent and Twofish.

Along with the announcement of the finalists, NIST published a status report [118] in which the selection was motivated. The choice coincided with the top five that resulted from the response to a questionnaire handed out at the end of the second AES workshop. Despite its moderate performance, Serpent made it thanks to its high security margin. The candidates that had not been eliminated because of security problems were not selected mainly for the following reasons:

1. CAST-256: comparable to Serpent but with a higher implementation cost.
2. Crypton: comparable to Rijndael and Twofish but with a lower security margin.
3. DFC: low security margin and bad performance on anything other than 64-bit processors.

4. E2: comparable to Rijndael and Twofish in structure, but with a lower security margin and higher implementation cost.
5. SAFER+: high security margin similar to Serpent but even slower.

1.7 The Second Round

After the announcement of the five candidates NIST made another open call for contributions focused on the finalists. Intellectual property issues and performance and chip area in dedicated hardware implementations entered the picture. A remarkable contribution originated from NSA, presenting the results of hardware-performance simulations performed for the finalists. This third AES conference was held in New York City in April 2000. As in the year before, it was combined with the Fast Software Encryption Workshop.

In the sessions on cryptographic attacks there were some interesting results but no breakthroughs, since none of the finalists showed any weaknesses that could jeopardize their security. Most of the results were attacks on reduced-round versions of the ciphers. All attacks presented are only of academic relevance in that they are only slightly faster than an exhaustive key search. In the sessions on software implementations, the conclusions of the second workshop were confirmed.

In the sessions on dedicated hardware implementations attention was paid to Field Programmable Gate Arrays (FPGAs) and Application-Specific Integrated Circuits (ASICs). In the papers Serpent came out as a consistently excellent performer. Rijndael and Twofish also proved to be quite suited for hardware implementation while RC6 turned out to be expensive due to its use of 32-bit multiplication. Dedicated hardware implementations of MARS seemed in general to be quite costly. The Rijndael-related results presented at this conference are discussed in more detail in Chap. 4 (which is on efficient implementations) and Chap. 10 (which is on cryptanalytic results).

At the end of the conference a questionnaire was handed out asking about the preferences of the attendants. Rijndael was resoundingly voted to be the public's favorite.

1.8 The Selection

On 2 October 2000, NIST officially announced that Rijndael, without modifications, would become the AES. NIST published an excellent 116-page report in which they summarize all contributions and motivate the choice [117]. In the conclusion of this report, NIST motivates the choice of Rijndael with the following words:

Rijndael appears to be consistently a very good performer in both hardware and software across a wide range of computing environments regardless of its use in feedback or non-feedback modes. Its key setup time is excellent, and its key agility is good. Rijndael's very low memory requirements make it very well suited for restricted-space environments, in which it also demonstrates excellent performance. Rijndael's operations are among the easiest to defend against power and timing attacks. Additionally, it appears that some defense can be provided against such attacks without significantly impacting Rijndael's performance.

Finally, Rijndael's internal round structure appears to have good potential to benefit from instruction-level parallelism.

2. Preliminaries

In this chapter we introduce a number of mathematical concepts and explain the terminology that we need in the specification of Rijndael (in Chap. 3), in the treatment of some implementation aspects (in Chap. 4) and when we discuss our design choices (Chaps. 5–9).

The first part of this chapter starts with a discussion of finite fields, the representation of their elements and the impact of this on their operations of addition and multiplication. Subsequently, there is a short introduction to linear codes. Understanding the mathematics is not necessary for a full and correct implementation of the cipher. However, the mathematics are necessary for a good understanding of our design motivations. Knowledge of the underlying mathematical constructions also helps for doing optimized implementations. Not all aspects will be covered in detail; where possible, we refer to books dedicated to the topics we introduce.

In the second part of this chapter we introduce the terminology that we use to indicate different common types of Boolean functions and block ciphers.

When the discussion moves from a general level to an example specific to Rijndael, the text is put in a grey box.

Notation. We use in this book two types of indexing:

subscripts: Parts of a larger, named structure are denoted with subscripts. For instance, the bytes of a state $\mathbf{a}$ are denoted by $a_{i,j}$ (see Chap. 3).

superscripts: In an enumeration of more or less independent objects, where the objects are denoted by their own symbols, we use superscripts. For instance the elements of a nameless set are denoted by $\{\mathbf{a}^{(1)}, \mathbf{a}^{(2)}, \dots\}$, and consecutive rounds of an iterative transformation are denoted by $\rho^{(1)}, \rho^{(2)}, \dots$ (see Sect. 2.3.4).

J. Daemen, V. Rijmen, *The Design of Rijndael*, Information Security and Cryptography,
https://doi.org/10.1007/978-3-662-60769-5_2

2.1 Finite Fields

In this section we present a basic introduction to the theory of finite fields. For a more formal and elaborate introduction, we refer to the work of Lidl and Niederreiter [95].

2.1.1 Groups, Rings and Fields

We start with the formal definition of a group.

Definition 2.1.1. *An* Abelian group $(G, +)$ *consists of a set* G *and an operation defined on its elements, here denoted by '*$+$*':*

$$+ : G \times G \to G : (a, b) \mapsto a + b. \tag{2.1}$$

For the group to qualify as an Abelian group*, the operation has to fulfill the following conditions:*

$$\begin{aligned}
\textit{closed:}\ & \forall\ a, b \in G : a + b \in G && (2.2)\\
\textit{associative:}\ & \forall\ a, b, c \in G : (a + b) + c = a + (b + c) && (2.3)\\
\textit{commutative:}\ & \forall\ a, b \in G : a + b = b + a && (2.4)\\
\textit{neutral element:}\ & \exists\, \mathbf{0} \in G, \forall\ a \in G : a + \mathbf{0} = a && (2.5)\\
\textit{inverse elements:}\ & \forall\ a \in G, \exists\ b \in G : a + b = \mathbf{0}. && (2.6)
\end{aligned}$$

Example 2.1.1. The best-known example of an Abelian group is $(\mathbb{Z}, +)$, the set of integers, with the operation 'addition'. The structure $(\mathbb{Z}_n, +)$ is a second example. The set contains the integer numbers 0 to $n - 1$ and the operation is addition modulo n.

Since the addition of integers is the best-known example of a group, usually the symbol '+' is used to denote an arbitrary group operation. In this book, both an arbitrary group operation and integer addition will be denoted by the symbol '+'.

Both rings and fields are formally defined as structures that consist of a set of elements with *two* operations defined on these elements.

Definition 2.1.2. *A* ring $(R, +, \cdot)$ *consists of a set* R *with two operations defined on its elements, here denoted by '*$+$*' and '*$\cdot$*'. For* R *to qualify as a ring, the operations have to fulfill the following conditions:*

1. *The structure* $(R, +)$ *is an Abelian group.*
2. *The operation '*$\cdot$*' is closed, and associative over* R*. There is a neutral element for '*$\cdot$*' in* R*.*

3. *The two operations '+' and '·' are related by the law of distributivity:*

$$\forall\ a, b, c \in R:\ (a + b) \cdot c = (a \cdot c) + (b \cdot c). \tag{2.7}$$

The neutral element for '·' is usually denoted by **1**. A ring $(R, +, \cdot)$ is called a *commutative ring* if the operation '·' is commutative.

Example 2.1.2. The best-known example of a ring is $(\mathbb{Z}, +, \cdot)$: the set of integers, with the operations 'addition' and 'multiplication'. This ring is a commutative ring. The set of matrices with n rows and n columns, with the operations 'matrix addition' and 'matrix multiplication' is a ring, but not a commutative ring (if $n > 1$).

Definition 2.1.3. *A structure $(F, +, \cdot)$ is a* field *if the following two conditions are satisfied:*

1. *$(F, +, \cdot)$ is a commutative ring.*
2. *For all elements of F, there is an inverse element in F with respect to the operation '·', except for the element* **0***, the neutral element of $(F, +)$.*

A structure $(F, +, \cdot)$ is a field iff both $(F, +)$ and $(F \backslash \{\mathbf{0}\}, \cdot)$ are Abelian groups and the law of distributivity applies. The neutral element of $(F \backslash \{\mathbf{0}\}, \cdot)$ is called the *unit element* of the field.

Example 2.1.3. The best-known example of a field is the set of real numbers, with the operations 'addition' and 'multiplication.' Other examples are the set of complex numbers and the set of rational numbers, with the same operations. Note that for these examples the number of elements is infinite.

2.1.2 Vector Spaces

Let $(F, +, \cdot)$ be a field, with unit element **1**, and let $(V, +)$ be an Abelian group. Let '$\odot$' be an operation on elements of F and V:

$$\odot:\ F \times V \to V. \tag{2.8}$$

Definition 2.1.4. *The structure $(F, V, +, +, \cdot, \odot)$ is a* vector space *over F if the following conditions are satisfied:*

1. *distributivity:*

$$\forall\ a \in F, \forall\ \mathbf{v}, \mathbf{w} \in V:\ a \odot (\mathbf{v} + \mathbf{w}) = (a \odot \mathbf{v}) + (a \odot \mathbf{w}) \tag{2.9}$$

$$\forall\ a, b \in F, \forall\ \mathbf{v} \in V:\ (a + b) \odot \mathbf{v} = (a \odot \mathbf{v}) + (b \odot \mathbf{v}). \tag{2.10}$$

2. *associativity:*

$$\forall\ a, b \in F, \forall\ \mathbf{v} \in V: (a \cdot b) \odot \mathbf{v} = a \odot (b \odot \mathbf{v}). \tag{2.11}$$

3. neutral element:

$$\forall\ \mathbf{v} \in V : \mathbf{1} \odot \mathbf{v} = \mathbf{v}. \tag{2.12}$$

The elements of V are called *vectors*, and the elements of F are the *scalars*. The operation '$+$' is called the *vector addition*, and '$\odot$' is the *scalar multiplication*.

Example 2.1.4. For any field F, the set of n-tuples $(a_0, a_1, \ldots, a_{n-1})$ forms a vector space, where '$+$' and '$\odot$' are defined in terms of the field operations:

$$(a_1, \ldots, a_n) + (b_1, \ldots, b_n) = (a_1 + b_1, \ldots, a_n + b_n) \tag{2.13}$$

$$a \odot (b_1, \ldots, b_n) = (a \cdot b_1, \ldots, a \cdot b_n). \tag{2.14}$$

A vector $\mathbf{v}$ is a *linear combination* of the vectors $\mathbf{w}^{(1)}, \mathbf{w}^{(2)}, \ldots, \mathbf{w}^{(s)}$ if there exist scalars $a^{(i)}$ such that:

$$\mathbf{v} = a^{(1)} \odot \mathbf{w}^{(1)} + a^{(2)} \odot \mathbf{w}^{(2)} + \ \cdots\ + a^{(s)} \odot \mathbf{w}^{(s)}. \tag{2.15}$$

In a vector space we can always find a set of vectors such that all elements of the vector space can be written in exactly one way as a linear combination of the vectors of the set. Such a set is called a *basis* of the vector space. We will consider only vector spaces where the bases have a finite number of elements. We denote a basis by a column vector e:

$$\mathsf{e} = \left[\mathbf{e}^{(1)}, \mathbf{e}^{(2)}, \ldots \mathbf{e}^{(n)}\right]^{\mathrm{T}}. \tag{2.16}$$

In this expression the T superscript denotes taking the transpose of the row vector in the right-hand side of the equation. The scalars used in this linear combination are called the *coordinates* of $\mathbf{x}$ with respect to the basis e:

$$\mathrm{co}(\mathbf{x}) = \boldsymbol{x} = (c_1, c_2, \ldots, c_n) \Leftrightarrow \mathbf{x} = \textstyle\sum_{i=1}^{n} c_i \odot \mathbf{e}^{(i)}. \tag{2.17}$$

In order to simplify the notation, from now on we will denote vector addition by the same symbol as the field addition ('$+$'), and the scalar multiplication by the same symbol as the field multiplication ('$\cdot$'). It should always be clear from the context what operation the symbols are referring to.

A function f is called *a linear function* of a vector space V over a field F, if it has the following properties:

$$\forall\ \mathbf{x}, \mathbf{y} \in V :\ f(\mathbf{x} + \mathbf{y}) = f(\mathbf{x}) + f(\mathbf{y}) \tag{2.18}$$

$$\forall\ a \in F, \forall\ \mathbf{x} \in V :\ f(a\mathbf{x}) = af(\mathbf{x}). \tag{2.19}$$

The linear functions of a vector space can be represented by a matrix multiplication on the coordinates of the vectors. A function f is a linear function of the vector space $\mathrm{GF}(p)^n$ iff there exists a matrix M such that

$$\mathrm{co}(f(\mathbf{x})) = \mathsf{M} \cdot \boldsymbol{x}, \forall\ \mathbf{x} \in \mathrm{GF}(p)^n. \tag{2.20}$$