

Pythonで学ぶ 暗号理論

博士(理学) 神永 正博 共著
博士(工学) 吉川 英機

コロナ社

まえがき

本書は、専門的に暗号理論を学ぶ大学生、大学院生、技術者向けの教科書である。もともと、暗号の研究は、軍事的な目的で秘密裏になされていた。1970年代の終わりに、標準暗号アルゴリズム DES や、公開鍵暗号が発明され、後にインターネットが普及してから暗号の研究が急速に拡大した。Wi-Fi、インターネットショッピング、IC乗車券、携帯電話のSIMカード、SNSでの暗号通信、ブロックチェーン技術など、いまや暗号なしの生活は考えられない。情報セキュリティ技術はソフトウェアを含む広大な領域であり、暗号理論はその一部にすぎないが、きわめて重要な領域である。暗号は、自己と他者を区別する免疫の役割を担っているからである。

暗号理論の核心部分は、共通鍵暗号、ハッシュ関数、公開鍵暗号の3つである。本書では、共通鍵暗号、ハッシュ関数、RSA暗号、ラビン暗号、楕円曲線暗号の解説とともに、ブロック暗号に対する差分解読法、線形解読法、そしてハッシュ関数の解析、RSA暗号に対する攻撃などを体験できるPythonプログラムを提供する。本書を活用することにより、理論の学習だけでは実感が湧きにくい暗号の仕組みについて、プログラムを動かしながら楽しく学べると考えている。

Pythonには、暗号処理用のモジュールが用意されており、これらを使えば、ほとんどのことができる。そのためには、そこで何が行われているかを理解しておく必要がある。また、IoT端末などでは、ハードウェア規模や電力リソースの制限などで、便利なモジュールを直接利用できないことも多く、そうした機器への暗号技術の実装のためには、原理面からの理解が欠かせない。本書では、暗号の解読技術を通じて「何が危険なのか」の説明に比較的多くのページを割いている。加えて、利用されている暗号のアルゴリズムについても、どのような難しさがあり、それがどう克服されているか、またはどのように現実と妥協しているかについて説明する。

暗号理論は、符号理論と並んで、実用的な技術の基礎理論としては際立って純粋数学的であり、特に代数学に関する知識が多用される。伝統的な工学教育では特に代数学の教育がきわめて限定的で、そのことが暗号理論を学ぶ際の大きな障害となっているように思われる。数学科で学ぶような代数学（群論、環論、体論、初等整数論）の基礎をすべて学んでから暗号理論を学ぶのは理想的だが、工学系の学生には負担が大きすぎるであろう。本書ではこうした知識を補い、Pythonで実際にアルゴリズムを実装して理解を深めていく。暗号で必要になる代数学の知識は多岐にわたるが、必要になった段階で導入する形を取った。

適宜、Pythonの操作やプログラムの実行などを含む基本的な問題が出題されているので、解いてみていただきたい。ごく一部を除いては簡単に解くことができるはずである。やや時間がかかりそうな問題には、*印がついている。時間がないときは*印のついている問題を飛ばして

読んでも問題なく読み進めることができるように配慮した。プログラムの実行には、Spyder、Visual Studio 等の統合環境を想定しているが、実習の際には、Jupyter Notebook や Google colab を使うほうが適しているかもしれない。読者にとって最も適当な環境で使ってもらえれば幸いである。

暗号学習者が陥りがちな罫は、暗号理論の一部のみの学習を深めてしまうことである。例えば、数学が達人な人は公開鍵暗号ばかり勉強してしまいがちであるが、それだけでは実用的なシステムをつくることは難しい。少なくとも、共通鍵暗号、ハッシュ関数、公開鍵暗号とその周辺技術、モジュールまで含めた理解とソフトウェア実装経験が必要であろう。

なお、本書を通じて、すでにモジュールが用意されていても、暗号の仕組みを理解するためにあえてプログラムを提供する場合がある。プログラムは練習問題の解答にあたるものも含めて、コロナ社のサイト (<https://www.coronasha.co.jp/np/isbn/9784339029468/>) より取得することができる。本書のプログラムはアルゴリズムを説明するためのものなので、説明の邪魔になる例外処理を含め情報セキュリティ上必要なパラメータのチェックなどは行っていない。そのまま製品やサービスに組み込まないようお願いするとともに、本書のプログラムは、あくまで暗号の基本原則を理解するためにのみお使いいただければ幸いである。多数のプログラムが提供されるが、何度も利用される処理をクラスにまとめるようなことはしていない。あちこちで重複が生じているが、アルゴリズム全体を理解するには、このほうが好都合だと思われる。

本書は、著者（神永）が、工学部の「情報セキュリティ工学」と大学院電気工学専攻における「暗号・情報セキュリティ工学特論」で毎年アップデートしながら講義してきた内容をベースにして執筆した。神永が全体設計を行い、共通鍵暗号・ハッシュ関数の解析を得意とする吉川が、第3章（の前半）、第4章、第6章を執筆し、両者で全体の見直し、表現の統一などを行ってまとめたものである。東北学院大学の森島佑先生には暗号利用モードの処理についてご教授いただいた。また、神永研究室の大学院生の佐藤健人君にもプログラムのチェックをしていただいた。記して感謝したい。

2024年8月

神永 正博, 吉川 英機

目 次

1. 共通鍵暗号

1.1 共通鍵暗号の基本	1
1.1.1 暗号化, 復号, 秘密鍵	1
1.1.2 チャレンジレスポンス認証	2
1.1.3 暗号用乱数と安全なパスワード生成	2
1.1.4 リレーアタック	4
1.1.5 暗号への攻撃法の種類	5
1.2 単換字式暗号と頻度解析	6

2. ブロック暗号の基礎

2.1 ブロック暗号の構成要素	12
2.2 写像の合成とブロック暗号	13
2.3 DES	15
2.4 DES の Python プログラムの詳細	19
2.4.1 線形変換	24
2.4.2 非線形変換 (S ボックス)	24
2.4.3 トリプル DES	25
2.4.4 DES-X	25

3. 現代のブロック暗号と暗号利用モード

3.1 ブロック暗号 AES	27
3.2 ブロック暗号 RC6	34
3.3 Pycryptodome モジュールと暗号利用モード	38
3.3.1 Pycryptodome モジュールの導入	38
3.3.2 暗号利用モードとデータの暗号化の実際	39
3.3.3 ECB モードの致命的な問題点	43

4. ブロック暗号に対する差分解読法・線形解読法

4.1	ブロック暗号 FEAL	45
4.2	FEAL に対する差分解読法	48
4.3	FEAL に対する線形解読法	52

5. ハッシュ関数とメッセージ認証子

5.1	ハッシュ関数とは何か	59
5.2	バースデーパラドックス	62
5.3	マークル=ダンガード構成	64
5.4	HMAC	67

6. ハッシュ関数の衝突シミュレーション

6.1	ハッシュ関数 MD4 の衝突を見つける	69
6.1.1	ハッシュ関数 MD4	69
6.1.2	MD4 における圧縮関数の性質	72
6.2	MD4 の衝突ペアの実例	73
6.3	MD4 の衝突ペアの導出法	74

7. RSA 暗号と RSA 電子署名

7.1	数学的準備	83
7.1.1	モジュラー算術	83
7.1.2	群の概念	84
7.1.3	既約剰余類群	88
7.2	RSA 暗号の原理	91
7.3	RSA 電子署名・ブラインド署名	93
7.4	ユークリッドの互除法・拡張ユークリッド互除法	94
7.5	RSA 暗号の実装例	98

8. RSA 暗号の実装アルゴリズム

8.1	べき乗剰余計算のアルゴリズム	101
8.2	中国人剰余定理	106
8.3	中国人剰余定理を使った RSA 復号処理の実装	108
8.4	RSA 電子署名と検証の実際	110
8.4.1	Pycryptodome モジュールを利用した RSA 電子署名	110
8.4.2	RSA-OAEP と安全性の階層	112

9. 素数生成

9.1	素数の分布	115
9.2	素数分布を見る	118
9.3	素数定理の証明の大まかな方針とリーマンゼータ関数	120
9.4	素数生成	123

10. RSA 暗号に対する攻撃

10.1	共通の公開モジュラスに対する攻撃	126
10.2	ブロードキャスト攻撃とその一般化	127
10.3	短い秘密指数に対する連分数攻撃	129
10.3.1	連分数展開と主近似分数	129
10.3.2	連分数攻撃の原理とシミュレーション	132

11. 平方剰余とラビン暗号

11.1	ラビン暗号	135
11.2	平方剰余	136
11.3	N の素因数がわかればラビン暗号が解読できる	141
11.4	モジュラー平方根の計算と法の素因数分解は同値である	142
11.5	ラビン暗号の Python 実装	143
11.6	トネリ=シャンクスアルゴリズム (発展事項)	145
11.7	平方剰余計算関数の Python 実装	146

12. 楕円曲線と楕円曲線上の離散対数問題

12.1	素 体	149
12.2	楕 円 曲 線	150
12.3	有 理 点 群	156
12.4	楕円曲線上のスカラー倍の実装	158
12.5	楕円曲線上の離散対数問題 (ECDLP)	161
12.6	有理点群の位数の計算	164

13. 楕円曲線の暗号への応用

13.1	楕円曲線ディフィー・ヘルマン鍵交換 (ECDH)	168
13.2	楕円曲線署名 (ECDSA)	170
13.3	ECDSAの実装	172
13.4	Pycryptodome による ECDSA	175
13.5	Dual_EC_DRBG のバックドア	177

引用・参考文献	181
練習問題略解	184
索 引	197



共通鍵暗号



本章では、共通鍵暗号の仕組みの概略、特に通信路暗号化とチャレンジレスポンス認証を説明し、素朴な暗号として、単換字式暗号とその解読方法である頻度解析を説明する。

1.1 共通鍵暗号の基本

1.1.1 暗号化、復号、秘密鍵

暗号理論で用いられる基本的な概念を説明しておこう。最もシンプルな共通鍵暗号のイメージは、**図 1.1** のようなものである。図 1.1 は、A 氏から B 氏にメッセージ P を暗号化して送る様子を表している。メッセージ（暗号化されていないもの）は、**平文** (plaintext) と呼ばれる。読みは、「ひらぶん」であって「へいぶん」ではない。平文が秘密鍵 K によって暗号化され、暗号文となる。共通鍵暗号では、秘密鍵 K は何らかの手段で通信前に共有されている必要がある。図 1.1 における E_K は暗号化関数であり、秘密鍵 K を用いて平文 P を暗号文 C に変換する。つまり、 $C = E_K(P)$ と書くことができる。一方、 D_K は復号関数である。 K を固定したとき、 D_K は逆関数 $D_K = E_K^{-1}$ になっている。つまり、 $D_K(C) = E_K^{-1}(E_K(P)) = P$ となる。 K を固定したときは、平文と暗号文は一対一に対応している。理想的な暗号では、 K を知らない限り、 C は乱数と区別できず、暗号文だけ見ても平文に関する情報が一切得られない。ただ、実際にはそうではないところが暗号理論の面白いところである。暗号解読を試みる者を攻撃者（アタッカー）という。

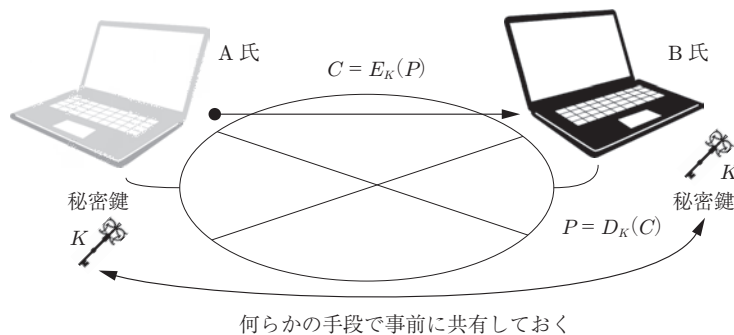


図 1.1 インターネットにおける共通鍵暗号化と復号のイメージ

1.1.2 チャレンジレスポンス認証

図 1.1 は、通信路暗号化という、最もわかりやすい暗号の使い方であるが、この他に重要な使い方として**チャレンジレスポンス認証** (challenge-response authentication) がある。図 1.2 は、IC 乗車券を例に取ったチャレンジレスポンス認証の概念図である。IC 乗車券は、カード内に IC チップがあり、そこで暗号の処理を行うことができる。改札からの電波による電磁誘導で電力を生み出し、IC チップに供給する非接触 IC カードである。まず、IC チップが改札に近づくと電磁誘導により IC チップが起動し、ゲートを開けるように要求する。これに対し、改札は、**チャレンジ** (challenge) と呼ばれる乱数 r を IC 乗車券に向けて送信する。IC 乗車券は、乗車券発行時に付与された秘密鍵 K を用いてチャレンジ r を暗号化し、 $r' = E_K(r)$ (**レスポンス** (response)) を改札に送り返す。改札側 (正確には改札につながっているサーバ) は、自らも保持している共通の鍵 K を用いてチャレンジを暗号化したものと r' が一致しているかどうかを確認する。これが一致していれば、カード側は発行者のリストにある鍵 K を保持していることがわかる。残高などが規定条件を満たせば、ゲートを開ける、という流れになる。チャレンジは毎回異なるので、 K を持っていない限り正しい $E_K(r)$ を計算することはできない。これがチャレンジレスポンス認証である^{†1}。チャレンジレスポンス認証の仕組みは幅広く用いられており、車の鍵として遠隔で開錠できるキーレスエントリーでも同様である。

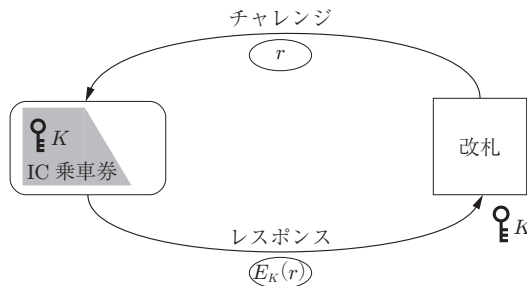


図 1.2 チャレンジレスポンス認証

1.1.3 暗号用乱数と安全なパスワード生成

なお、一般にチャレンジは**疑似乱数** (pseudorandom numbers) である。本物の乱数を疑似乱数に対比させて**真正乱数** (true random numbers) と呼ぶ。真正乱数に近いものがよい疑似乱数であるが、その「近さ」に関しては、少なくとも 2 つの視点がある。1 つは統計的な視点である。例えば、0 と 1 の頻度や、独立性などを統計的な検定で評価し、多くの検定を通過するものがよい疑似乱数である。この方向では、**メルセンヌツイスタ** (Mersenne Twister, MT) がきわめて優れた疑似乱数生成器である^{†2}。MT は、松本眞と西村拓士によって発明された。

^{†1} r はメッセージにあたるが、乱数であって、何か意味のある情報ではないことに注意されたい。

^{†2} より高速な SIMD-oriented Fast Mersenne Twister (SFMT), Double precision SIMD-oriented Fast Mersenne Twister (dSFMT) もあり、現在広く利用されている。

Matsumoto-Nishimura¹⁾は記念碑的論文である（論文は1998年掲載だが、結果は1996年の国際会議で発表されている）。MTは、 $2^{19937} - 1$ （メルセンヌ素数）という長大な周期を持っており、モンテカルロシミュレーション（乱数を用いたシミュレーション）などではほとんどの場合満足のいく乱数を生成する。Pythonをはじめとして、多くのプログラム言語が、MTを採用して乱数を生成している。Pythonの`random`モジュールはMTを提供している。もう1つの視点は、予測困難性である。疑似乱数は所定の規則に従ってシード（初期値）を更新するアルゴリズムであるから、原理的には予測可能だ。しかし、暗号用途では、予測困難であることが必要である。MTは、予測困難ではないので、暗号用途で使うことは避けなければならない。MTのコード²⁾によれば、MTは内部ステートとして624個の32ビットの（unsigned long型）整数を持っており、624回出力すると内部ステートがすべて更新される。したがって、624個の出力からMTの内部ステートを復元してつぎの出力が予測できる。

Pythonで暗号用の乱数を生成したい場合は、`secrets`モジュール³⁾を利用するのが一般的である。例えば、`secrets.randbits(k)`は k ビットの予測困難な疑似乱数を生成する。なお、Cに標準装備されている`rand`は周期の短い線形合同法で生成された疑似乱数を提供するが、統計的視点で見てもまったく不十分なものであり、予測困難性の観点では論外である。

真正乱数は真円と同様に数学における理想的な概念であるが、暗号理論の文脈では、物理的な手段で生成される乱数を真正乱数と呼ぶことが多い。例えば、抵抗間の電圧のゆらぎをオペアンプで増幅し、そのアナログ信号をビット列に変換したり、高速なクロック信号を低速なクロックでサンプリングするなどの方法が用いられる。他にも環境電磁波を利用するなどの方法もある。これらは、温度や電圧状況を故意に操作した場合に乱数の質が悪化する可能性があるが、通常は、物理乱数とよい疑似乱数生成器の組み合わせで十分安全な乱数が生成できると考えられている。

問題 1-1 `secrets`モジュールの`secrets.randbits(k)`を用いて、8ビットの乱数を生成し、2進数表示せよ。2進数表示するには、`bin`関数を用いればよい。

`secrets`モジュールを使えば安全なパスワードを生成するプログラムも簡単にできる。リスト1.1のプログラムは、`length`で指定した文字数（ここでは8）のランダムなパスワードを生成するものである。

リスト 1.1 (StudySecurePassword.py)

```
1 import secrets
2 import string
3
4 length = 8
5 lst = string.ascii_letters + string.digits
6 password = ''.join(secrets.choice(lst) for i in range(length))
```

¹⁾ 肩付きの数字は、巻末の引用・参考文献の番号を示す。

²⁾ <http://www.math.sci.hiroshima-u.ac.jp/m-mat/MT/MT2002/CODES/mt19937ar.c>

³⁾ このモジュールは、オペレーティングシステム（OS）が提供する最も安全な乱数源に基づく乱数を生成するために用いられる。`secrets`は、`os.urandom`と`random.SystemRandom`に基づく。

```
7 print("password:", password)
```

`string` モジュールは、アスキー文字列、数字列などを含むもので、5行目の `lst` は、アルファベットと数字を列挙したもの、つまり

```
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'
```

である。6行目で `secrets.choice` を使って、上記の文字列から、`length` で指定した回数だけランダム選択してつないでパスワードを生成している。

問題 1-2 リスト 1.1 のプログラムを修正して、10 文字のパスワードを生成してみよ。

1.1.4 リレーアタック

一見すると暗号を破ることなしにチャレンジレスポンス認証を破ることは不可能に思えるが、実際には抜け道がある。実際に犯罪に使われたものとして自動車のスマートキーに対する **リレーアタック** (vehicle relay attack) がある。リレーアタックは、スマートキーが車に近づくと (1 m 程度) 鍵が開くという仕組みを利用した攻撃手法である。この物理的に鍵を差し込まずに解錠できるシステムは、**スマートエントリーシステム** (smart-entry system) と呼ばれるもので、1993 年にシボレー・コルベットに採用された **パッシブ・キーレスエントリー・システム** (Passive Keyless Entry System) や 1998 年にメルセデス・ベンツ・S クラスに採用された「キーレス・ゴー」など広く用いられている。スマートエントリーシステムでは、ドライバーが車から離れると自動的に施錠される。

車の盗難防止の仕組みとしては、**イモビライザ** (immobilizer) がある。イモビライザとは、電子的なキーの照合システムによって、専用のキー以外ではエンジンの始動ができないという自動車盗難防止システムのことである。物理的な方法でドアを開けてもエンジンがかからないという意味で効果的な盗難対策である。スマートキーと呼ばれるものは通常イモビライザキーを兼ねている。

リレーアタックでは、スマートエントリーにおいて、スマートキーが近くにあることを車のコントロールユニットが検知して (弱い電波が発信されている) ドアロックが解除され、エンジンがかかる (イモビライザを兼ねているから) ことを利用して車を盗難する。スマートキーが近くにあるのと等価な状況をつくり出すことができれば施錠を解くことができるからである。一方の機器は車の隣に配置する必要があり、もう一方の機器は車の所有者のスマートキーの近くに配置する必要がある。

リレーアタックでは、自動車とスマートキーを持っているドライバーとの間に入り、アンプで信号を増幅して中継することでロックの解除が可能になる。信号は復号されるのではなくコピーされ、結果的には秘密鍵を傍受するのと同じ効果がある。リレーアタックによる車の盗難は、欧米で先行して確認され、2018、2019 年頃には日本でも被害が確認された。リレーアタックの手順をまとめるとつぎのようになる。犯行は通常複数名で行われることが多い (図 1.3)。

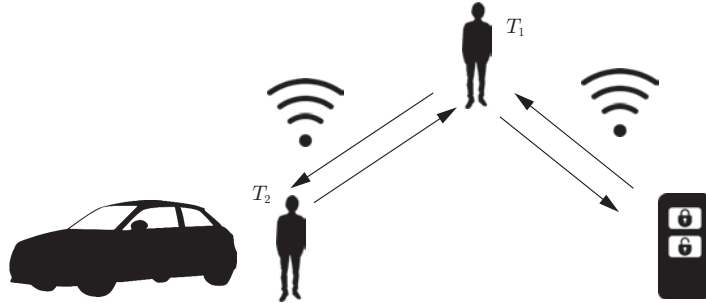


図 1.3 リレーアタック

1. グループの 1 人 T_1 がスマートキーに近づく（キーは運転手自身の手元にある場合もあるし、そうでない場合もある。また、運転手の手元ではなく、例えば、玄関先にキーが置かれている場合であれば、玄関に近づくという意味）
2. スマートキーの微弱電波を受信し増幅させて犯行グループの別の一人 T_2 に送信
3. 増幅させた電波を受けた T_2 が受信機を持って車に近づき、ロック解除
4. エンジン始動・盗難

リレーアタックにおいては、犯人は暗号を解読したわけではない。彼らは信号を中継しただけである。これは、**中間者攻撃** (man-in-the-middle attack) の一種である[†]。プロトコルを変えずに対策するには、スマートキーを金属缶などで遮蔽^{しやへい}することが比較的容易で有効な対策である。ポータブルな遮蔽グッズも販売されている。IC 乗車券などでリレーアタックが問題にならないのは、電波の中継が困難だからである。

1.1.5 暗号への攻撃法の種類

暗号に対する攻撃はアタッカーがどのような情報を得ることができるかによって分類されることが多い。暗号においては秘密鍵を復元する攻撃が最も強力である。秘密鍵が手に入れば、どんな暗号文でも復号できるからである。暗号理論においては、平文 P 、暗号文 C 、秘密鍵 K についての方程式

$$E_K(P) = C$$

を何らかの方法で「解く」ことが求められる。この観点で暗号への攻撃を分類すると以下のようになる。

- **選択平文攻撃** (chosen plaintext attack)：アタッカーが、任意の平文 P に対し、暗号文 C を得ることができる場合の攻撃
- **既知平文攻撃** (known-plaintext attack)：アタッカーが、既知の平文 P に対応する暗号文 C を得ることができる場合の攻撃

[†] 通常の中間者攻撃では、A 氏、B 氏がおのおの C 氏を通信相手と思っている (A 氏は C 氏を B 氏と誤っており、B 氏も C 氏を A 氏と誤っている) 状況 (なりすまし) をつくり出さなければならない。

索引

【あ】		逆元	85	衝突困難性	59
圧縮関数	64	既約剰余類	88	剰余類	84
アノマラス曲線	175	既約剰余類群	89	初期化ベクトル	40
アーベル群	85	強衝突耐性	59	ショートパッド攻撃	129
暗号利用モード	38	【く】		シングル DES	25
【い】		繰り返し型ハッシュ関数群	67 85	真正乱数	2
位数	85, 86	【け】		伸長攻撃	66
一般化フェイステル構造	35	決定ディフィー・ヘルマン仮定	169	【す】	
イモビライザ	4	原始根	136	スカラー倍	158
【え】		原像計算困難性	59	ステート	29
エドワーズ曲線	176	【こ】		スマートエントリシステム	4
【お】		公開鍵証明書	114	【せ】	
オイラー関数	89	公開鍵ビニング	114	正規表現	7
【か】		公開指数	91	生成元	86
カウンタモード	40	公開モジュラス	91	セッション鍵	93
換字	12	恒等写像	13	線形式	53
可換環	84	コファクター	177	線形性	24
可換群	85	【さ】		線形変換	12, 24
鍵交換	92	最小公倍数	95	全射	13
鍵スケジュール	15, 22	最大公約数	95	選択平文攻撃	5
鍵つきハッシュ	67	差分	48	全単射	13
鍵導出関数	170	算術級数の素数定理	118	【そ】	
可逆	88	【し】		総当たり攻撃	6
拡張ユークリッド互除法	95, 97	事後漂白	25	素数生成	123
拡張リーマン仮説	146	指数計算法	167	素体	149
確定的暗号系	112	事前計算	103	ソロベイ・シュトラッセンテスト	125
確定的素数判定法	123	事前漂白	25	【た】	
確率的暗号系	112	弱衝突耐性	60	体	149
確率的素数判定法	123	写像	13	対数積分関数	119
可算	156	シャンクス=メッスルのアルゴリズム	164	大数の法則	9
ガーナーの公式	109	主近似分数	131	第二原像計算困難性	59
カーマイケル数	125	出力差分	48	代表元	84
ガロア体	149	シューフのアルゴリズム	164	楕円曲線	150
環	84	巡回群	86	楕円曲線署名	170
【き】		準同型写像	87	楕円曲線ディフィー・ヘルマン鍵交換	168
疑似乱数	2	衝突	59	たがいに素	95
既知平文攻撃	5			多項分布	9
基底簡約	134			単位元	85

単換字式暗号	6	ハイブリッド暗号システム	93		
単射	13	パースデーパラドックス	62		
		パッシブ・キーレスエントリー・システム	4	【み】	
【ち】		ハッシュ関数	59	ミラー・ラビンテスト	124
チャレンジ	2	ハッシュ値	59	【む】	
チャレンジレスポンス認証	2	パブリックナンス	180	無限遠点	156
中間者攻撃	5	パリティ関数	59	無限群	85
中国人剰余定理	106			【め】	
超特異曲線	175			メッセージダイジェスト	59
直積	14, 87	【ひ】		メルセンヌツイスタ	2
直積集合	14	非線形性	24	【や】	
		非線形変換	24	ヤコビ記号	138
【つ】		ビット長	7	【ゆ】	
対合	17	秘密指数	91	有限群	85
強い嘘つき	124	秘密素数	91	有限体	149
強い擬素数	124	漂白	25	有理点	152
		平文	1	有理点群	152, 157
【て】				ユークリッド互除法	95
ディオファントス近似論	130	【ふ】		【ら】	
ディフィー・ヘルマン仮定	169	フェイステル型ブロック暗号	16	ラウンド鍵	15
デジタル証明書	114	フェルマーの小定理	91	ラウンド関数	15
テストベクタ	18	フェルマー法	125	ランダムオラクル	113
転置	12	不動点	15	ランダムオラクルモデル	113
		部分群	85	【り】	
【と】		ブラインド署名	94	離散対数問題	161
同型	88	ブロック	12	リーマンゼータ関数	122
トネリ=シャンクスアルゴリズム	145	ブロックサイズ	12	リレーアタック	4
トリプル DES	25	ブロードキャスト攻撃	127	【る】	
トレース	154			ルジャンドル記号	137
		【へ】		【れ】	
【な】		米国立標準技術研究所	16	レスポンス	2
ナンス	41	平方剰余	136	連結演算子	28
		平方非剰余	136	連分数攻撃	129
【に】		ベースポイント	168	連分数展開	130
二進体	149	【ほ】		【わ】	
入力差分	48	補助鍵	15	ワイエルシュトラス標準形	150
認証局	114	ポーリグ=ヘルマン攻撃	175		
		【ま】			
【の】		マークル=ダンガード強化法	66		
ノンス	41	マークル=ダンガード構成	65		
		マスク生成関数	113		
【は】					
バイナリ法	101				

【A】

AES	14
AKS 素数判定法	125
AONT	113

【B】

BSGS アルゴリズム	164
-------------	-----

【C】

CBC メッセージ認証子	40
CBC-MAC	40
CRT 係数	109

CRT 再結合	106			RSA 電子署名	94
CRT-RSA	109		[H]		
CTR モード	40	HKDF			
		HMAC		[S]	
[D]				S ボックス	12
der 形式	111		[L]	SP ネットワーク	28
DES	15	LLL 基底簡約アルゴリズム		SPN 型ブロック暗号	28
DES-X	25			SSSA 攻撃	175
Dual_EC_DRBG	178		[M]		
		MOV 帰着		[T]	
[E]				TLS	100
ECB モード	39		[N]	Tor	100
ECDH	169	NIST			
ECDLP	161			[V]	
ECDSA	170		[O]	VoIP	100
		OAEP		VPN	100
[F]				[数字]	
FEAL	45		[P]	2^k -ary 法	103
		pem 形式		2 鍵トリプル DES	25
[G]		PKCS#7 パディング		2 倍算	156
g_1, g_2, \dots, g_m で生成される部分群	86		[R]	3 鍵トリプル DES	25
		Rijndael			
					27

—— 著者略歴 ——

神永 正博 (かみなが まさひろ)

1991年 東京理科大学理学部数学科卒業
1993年 京都大学大学院理学研究科修士課程修了(数学専攻)
1994年 京都大学大学院理学研究科博士課程中退(数学専攻)
1994年 東京電機大学助手
1998年 株式会社日立製作所勤務
2003年 博士(理学)(大阪大学)
2004年 東北学院大学講師
2005年 東北学院大学助教授
2007年 東北学院大学准教授
2011年 東北学院大学教授
現在に至る

吉川 英機 (よしかわ ひでき)

1989年 大分工業高等専門学校電気工学科卒業
1991年 電気通信大学電気通信学部通信工学科卒業
1993年 電気通信大学大学院博士前期課程修了
(電子情報学専攻)
1993年 鈴鹿工業高等専門学校助手
2000年 博士(工学)(大阪市立大学)
2003年 鈴鹿工業高等専門学校講師
2007年 東北学院大学准教授
2019年 東北学院大学教授
現在に至る

Python で学ぶ暗号理論

Cryptography with Python

© Masahiro Kaminaga, Hideki Yoshikawa 2024

2024年10月15日 初版第1刷発行

★

検印省略

著者 神永正博
吉川英機
発行者 株式会社 コロナ社
代表者 牛来真也
印刷所 三美印刷株式会社
製本所 株式会社 グリーン

112-0011 東京都文京区千石 4-46-10
発行所 株式会社 コロナ社
CORONA PUBLISHING CO., LTD.
Tokyo Japan
振替 00140-8-14844・電話(03)3941-3131(代)
ホームページ <https://www.coronasha.co.jp>

ISBN 978-4-339-02946-8 C3055 Printed in Japan

(西村)



JCOPY <出版者著作権管理機構 委託出版物>

本書の無断複製は著作権法上での例外を除き禁じられています。複製される場合は、そのつど事前に、出版者著作権管理機構(電話 03-5244-5088, FAX 03-5244-5089, e-mail: info@jcopy.or.jp)の許諾を得てください。

本書のコピー、スキャン、デジタル化等の無断複製・転載は著作権法上での例外を除き禁じられています。購入者以外の第三者による本書の電子データ化及び電子書籍化は、いかなる場合も認めていません。落丁・乱丁はお取替えいたします。