RC4 Encryption Algorithm

The RC4 Encryption Algorithm, developed by Ronald Rivest of [RSA](https://www.vocal.com/cryptography/rsa/), is a shared key stream cipher algorithm requiring a secure exchange of a shared key.**RC4 is no longer considered secure and careful consideration should be taken regarding it’s use.** The symmetric key algorithm is used identically for encryption and decryption such that the data stream is simply XORed with the generated key sequence. The algorithm is serial as it requires successive exchanges of state entries based on the key sequence. Hence implementations can be very computationally intensive. The RC4 encryption algorithm is used by standards such as IEEE 802.11 within WEP (Wireless Encryption Protocol) using 40 and 128-bit keys. Published procedures exist for cracking the security measures as implemented in WEP.

* [IEEE 802.11](https://www.vocal.com/networking/ieee802-11/)
* [Wireless Encryption Protocol](https://www.vocal.com/secure-communication/wired-equivalent-privacy-wep/)
* [Communications Software](https://www.vocal.com/software-modules/communication-security-software/)
* [Communications Design](https://www.vocal.com/resources/research/communication-security/)
* [Secure Reference Designs](https://www.vocal.com/resources/reference-designs/communication-security/)

RC4 Algorithm

In the RC4 encryption algorithm, the key stream is completely independent of the plaintext used. An 8 \* 8 S-Box (S0 S255), where each of the entries is a permutation of the numbers 0 to 255, and the permutation is a function of the variable length key. There are two counters i, and j, both initialized to 0 used in the algorithm.

The algorithm uses a variable length key from 1 to 256 bytes to initialize a 256-byte state table. The state table is used for subsequent generation of pseudo-random bytes and then to generate a pseudo-random stream which is XORed with the plaintext to give the ciphertext. Each element in the state table is swapped at least once.

The key is often limited to 40 bits, because of export restrictions but it is sometimes used as a 128 bit key. It has the capability of using keys between 1 and 2048 bits. RC4 is used in many commercial software packages such as Lotus Notes and Oracle Secure SQL.

The algorithm works in two phases, key setup and ciphering. Key setup is the first and most difficult phase of this encryption algorithm. During a N-bit key setup (N being your key length), the encryption key is used to generate an encrypting variable using two arrays, state and key, and N-number of mixing operations. These mixing operations consist of swapping bytes, modulo operations, and other formulas. A modulo operation is the process of yielding a remainder from division. For example, 11/4 is 2 remainder 3; therefore eleven mod four would be equal to three.

Strengths of RC4

* The difficulty of knowing where any value is in the table.
* The difficulty of knowing which location in the table is used to select each value in the sequence.
* Encryption is about 10 times faster than DES.

Limitations of RC4

* RC4 is no longer considered secure.
* One in every 256 keys can be a weak key. These keys are identified by cryptanalysis that is able to find circumstances under which one of more generated bytes are strongly correlated with a few bytes of the key.
* A particular RC4 Algorithm key can be used only once.

Performance

* Each of the UDI implementations is a hardware block specifically designed for the implementation. RAM space is required by the key byte generator to locally maintain the state table for key generation. This state would need to be preserved and restored in case of a context switch if other processes would need the same functionality. This overhead is not considered in the above performance projections. Encryption and decryption state data may be stored in separate state memories to allow for independent processes.
* The following table summarizes the number of MIPS required for the algorithm encryption/decryption for 1 million bits per second for each of the three implementations.

|  |  |  |
| --- | --- | --- |
|  | **MIPS** | **RAM** |
| **Optimized MIPS Assembly** | 2.5 | none |
| **RC4 Operation Support UDI Primitives** | 1.75 | 0 bytes |
| **RC4 Key Byte Generator UDI Accelerator** | 0.22 | 256 bytes |

RC4 Software

The VOCAL implementation of the RC4 algorithm is available in several forms. The forms include pure optimized software and varying levels of hardware complexity utilizing UDI instructions for improved performance. When special assistance hardware is not available (as is the case on most general purpose processors), the byte manipulation/exchange operations are implemented via software.

Academically speaking, RC4 is terrible; it has easy distinguishers ("easy" means "can really be demonstrated in lab conditions"). It is also hard to use properly. However, SSL/TLS uses RC4 correctly, and **in practice** the shortcomings of RC4 have no real importance.

The power-that-be at Google decided to switch to RC4 by default because of the recent "BEAST" attack, which demonstrates (again, in lab conditions) a compromise of a Paypal cookie. There is no such dramatic demonstration for an attack on RC4 as used in SSL, so it was estimated that using AES-CBC with SSL/TLS 1.0 was "more risky" than using RC4.

The academically "right thing" to do would be to use AES-CBC with TLS **1.1** (or any ulterior version), which has no problem with BEAST and none of the RC4-related weaknesses either. However, Google makes money in the real world, and, as such, they cannot enforce a configuration which would prevent a third of their user base from connecting.

RC4

From Wikipedia, the free encyclopedia

[Jump to navigation](https://en.wikipedia.org/wiki/RC4#mw-head)[Jump to search](https://en.wikipedia.org/wiki/RC4#searchInput)

*This article is about the stream cipher. For other uses, see*[*RC4 (disambiguation)*](https://en.wikipedia.org/wiki/RC4_(disambiguation))*.*

|  |  |
| --- | --- |
| **RC4** | |
| **General** | |
| **Designers** | [Ron Rivest](https://en.wikipedia.org/wiki/Ron_Rivest) ([RSA Security](https://en.wikipedia.org/wiki/RSA_Security)) |
| **First published** | Leaked in 1994 (designed in 1987) |
| **Cipher detail** | |
| [**Key sizes**](https://en.wikipedia.org/wiki/Key_size) | 40–2048 bits |
| **State size** | 2064 bits (1684 effective) |
| **Rounds** | 1 |
| **Speed** | 7 cycles per byte on [original Pentium](https://en.wikipedia.org/wiki/Intel_P5_(microarchitecture))[[1]](https://en.wikipedia.org/wiki/RC4#cite_note-1) Modified Alleged RC4 on Intel Core 2: 13.9 cycles per byte[[2]](https://en.wikipedia.org/wiki/RC4#cite_note-2) |

In [cryptography](https://en.wikipedia.org/wiki/Cryptography), **RC4** (Rivest Cipher 4 also known as **ARC4** or **ARCFOUR** meaning Alleged RC4, see below) is a [stream cipher](https://en.wikipedia.org/wiki/Stream_cipher). While it is remarkable for its simplicity and speed in software, multiple vulnerabilities have been discovered in RC4, rendering it insecure.[[3]](https://en.wikipedia.org/wiki/RC4#cite_note-rfc7465-3)[[4]](https://en.wikipedia.org/wiki/RC4#cite_note-4) It is especially vulnerable when the beginning of the output [keystream](https://en.wikipedia.org/wiki/Keystream) is not discarded, or when nonrandom or related keys are used. Particularly problematic uses of RC4 have led to very insecure [protocols](https://en.wikipedia.org/wiki/Cryptographic_protocol) such as [WEP](https://en.wikipedia.org/wiki/Wired_Equivalent_Privacy).[[5]](https://en.wikipedia.org/wiki/RC4#cite_note-5)

As of 2015, there is speculation that some state cryptologic agencies may possess the capability to break RC4 when used in the [TLS protocol](https://en.wikipedia.org/wiki/Transport_Layer_Security).[[6]](https://en.wikipedia.org/wiki/RC4#cite_note-Leyden20130906-6) [IETF](https://en.wikipedia.org/wiki/IETF) has published [RFC 7465](https://tools.ietf.org/html/rfc7465) to prohibit the use of RC4 in TLS;[[3]](https://en.wikipedia.org/wiki/RC4#cite_note-rfc7465-3) [Mozilla](https://en.wikipedia.org/wiki/Mozilla) and [Microsoft](https://en.wikipedia.org/wiki/Microsoft) have issued similar recommendations.[[7]](https://en.wikipedia.org/wiki/RC4#cite_note-7)[[8]](https://en.wikipedia.org/wiki/RC4#cite_note-8)

A number of attempts have been made to strengthen RC4, notably Spritz, RC4A, [VMPC](https://en.wikipedia.org/wiki/Variably_Modified_Permutation_Composition), and RC4+.



**Contents**

* [1History](https://en.wikipedia.org/wiki/RC4#History)
* [2Description](https://en.wikipedia.org/wiki/RC4#Description)
  + [2.1Key-scheduling algorithm (KSA)](https://en.wikipedia.org/wiki/RC4#Key-scheduling_algorithm_(KSA))
  + [2.2Pseudo-random generation algorithm (PRGA)](https://en.wikipedia.org/wiki/RC4#Pseudo-random_generation_algorithm_(PRGA))
  + [2.3RC4-based random number generators](https://en.wikipedia.org/wiki/RC4#RC4-based_random_number_generators)
  + [2.4Implementation](https://en.wikipedia.org/wiki/RC4#Implementation)
  + [2.5Test vectors](https://en.wikipedia.org/wiki/RC4#Test_vectors)
* [3Security](https://en.wikipedia.org/wiki/RC4#Security)
  + [3.1Roos's biases and key reconstruction from permutation](https://en.wikipedia.org/wiki/RC4#Roos's_biases_and_key_reconstruction_from_permutation)
  + [3.2Biased outputs of the RC4](https://en.wikipedia.org/wiki/RC4#Biased_outputs_of_the_RC4)
  + [3.3Fluhrer, Mantin and Shamir attack](https://en.wikipedia.org/wiki/RC4#Fluhrer,_Mantin_and_Shamir_attack)
  + [3.4Klein's attack](https://en.wikipedia.org/wiki/RC4#Klein's_attack)
  + [3.5Combinatorial problem](https://en.wikipedia.org/wiki/RC4#Combinatorial_problem)
  + [3.6Royal Holloway attack](https://en.wikipedia.org/wiki/RC4#Royal_Holloway_attack)
  + [3.7Bar-mitzvah attack](https://en.wikipedia.org/wiki/RC4#Bar-mitzvah_attack)
  + [3.8NOMORE attack](https://en.wikipedia.org/wiki/RC4#NOMORE_attack)
* [4RC4 variants](https://en.wikipedia.org/wiki/RC4#RC4_variants)
  + [4.1RC4A](https://en.wikipedia.org/wiki/RC4#RC4A)
  + [4.2VMPC](https://en.wikipedia.org/wiki/RC4#VMPC)
  + [4.3RC4+](https://en.wikipedia.org/wiki/RC4#RC4+)
  + [4.4Spritz](https://en.wikipedia.org/wiki/RC4#Spritz)
* [5RC4-based protocols](https://en.wikipedia.org/wiki/RC4#RC4-based_protocols)
* [6See also](https://en.wikipedia.org/wiki/RC4#See_also)
* [7References](https://en.wikipedia.org/wiki/RC4#References)
* [8Further reading](https://en.wikipedia.org/wiki/RC4#Further_reading)
* [9External links](https://en.wikipedia.org/wiki/RC4#External_links)

History[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=1)]

RC4 was designed by [Ron Rivest](https://en.wikipedia.org/wiki/Ron_Rivest) of [RSA Security](https://en.wikipedia.org/wiki/RSA_Security) in 1987. While it is officially termed "Rivest Cipher 4", the RC acronym is alternatively understood to stand for "Ron's Code"[[9]](https://en.wikipedia.org/wiki/RC4#cite_note-9) (see also [RC2](https://en.wikipedia.org/wiki/RC2), [RC5](https://en.wikipedia.org/wiki/RC5) and [RC6](https://en.wikipedia.org/wiki/RC6)).

RC4 was initially a [trade secret](https://en.wikipedia.org/wiki/Trade_secret), but in September 1994 a description of it was anonymously posted to the [Cypherpunks](https://en.wikipedia.org/wiki/Cypherpunk" \o "Cypherpunk) mailing list.[[10]](https://en.wikipedia.org/wiki/RC4#cite_note-10) It was soon posted on the [sci.crypt](https://en.wikipedia.org/wiki/Sci.crypt" \o "Sci.crypt) [newsgroup](https://en.wikipedia.org/wiki/Newsgroup), where it was analyzed within days by Bob Jenkins.[[11]](https://en.wikipedia.org/wiki/RC4#cite_note-11) From there it spread to many sites on the Internet. The leaked code was confirmed to be genuine as its output was found to match that of proprietary software using licensed RC4. Because the algorithm is known, it is no longer a trade secret. The name *RC4* is trademarked, so RC4 is often referred to as *ARCFOUR* or *ARC4* (meaning *alleged RC4*)[[12]](https://en.wikipedia.org/wiki/RC4#cite_note-12) to avoid trademark problems. [RSA Security](https://en.wikipedia.org/wiki/RSA_Security) has never officially released the algorithm; Rivest has, however, linked to the [English Wikipedia](https://en.wikipedia.org/wiki/English_Wikipedia) article on RC4 in his own course notes in 2008[[13]](https://en.wikipedia.org/wiki/RC4#cite_note-13) and confirmed the history of RC4 and its code in a 2014 paper by him.[[14]](https://en.wikipedia.org/wiki/RC4#cite_note-Rivest2014-14)

RC4 became part of some commonly used encryption protocols and standards, such as [WEP](https://en.wikipedia.org/wiki/Wired_Equivalent_Privacy) in 1997 and [WPA](https://en.wikipedia.org/wiki/Wi-Fi_Protected_Access) in 2003/2004 for wireless cards; and [SSL](https://en.wikipedia.org/wiki/Secure_Sockets_Layer) in 1995 and its successor [TLS](https://en.wikipedia.org/wiki/Transport_Layer_Security) in 1999, until it was prohibited for all versions of TLS by [RFC 7465](https://tools.ietf.org/html/rfc7465) in 2015, due to the [RC4 attacks](https://en.wikipedia.org/wiki/Transport_Layer_Security#RC4_attacks) weakening or breaking RC4 used in SSL/TLS. The main factors in RC4's success over such a wide range of applications have been its speed and simplicity: efficient implementations in both software and hardware were very easy to develop.

Description[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=2)]

RC4 generates a [pseudorandom stream of bits](https://en.wikipedia.org/wiki/Pseudo-random_number_generator) (a [keystream](https://en.wikipedia.org/wiki/Keystream)). As with any stream cipher, these can be used for encryption by combining it with the plaintext using bit-wise [exclusive-or](https://en.wikipedia.org/wiki/Exclusive_or); decryption is performed the same way (since exclusive-or with given data is an [involution](https://en.wikipedia.org/wiki/Involution_(mathematics))). This is similar to the [one-time pad](https://en.wikipedia.org/wiki/One-time_pad) except that generated *pseudorandom bits*, rather than a prepared stream, are used.

To generate the keystream, the cipher makes use of a secret internal state which consists of two parts:

1. A [permutation](https://en.wikipedia.org/wiki/Permutation) of all 256 possible [bytes](https://en.wikipedia.org/wiki/Bytes) (denoted "S" below).
2. Two 8-bit index-pointers (denoted "i" and "j").

The permutation is initialized with a variable length [key](https://en.wikipedia.org/wiki/Key_(cryptography)), typically between 40 and 2048 bits, using the [*key-scheduling*](https://en.wikipedia.org/wiki/Key_schedule) algorithm (KSA). Once this has been completed, the stream of bits is generated using the *pseudo-random generation algorithm* (PRGA).

**Key-scheduling algorithm (KSA)**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=3)]

The [key-scheduling](https://en.wikipedia.org/wiki/Key_schedule) algorithm is used to initialize the permutation in the array "S". "keylength" is defined as the number of bytes in the key and can be in the range 1 ≤ keylength ≤ 256, typically between 5 and 16, corresponding to a [key length](https://en.wikipedia.org/wiki/Key_length) of 40 – 128 bits. First, the array "S" is initialized to the [identity permutation](https://en.wikipedia.org/wiki/Identity_permutation). S is then processed for 256 iterations in a similar way to the main PRGA, but also mixes in bytes of the key at the same time.

**for** i **from** 0 **to** 255

S[i] := i

**endfor**

j := 0

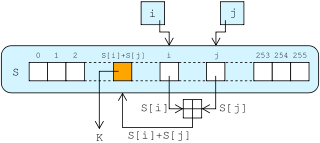
**for** i **from** 0 **to** 255

j := (j + S[i] + key[i [mod](https://en.wikipedia.org/wiki/Modulo_operation) keylength]) mod 256

swap values of S[i] and S[j]

**endfor**

**Pseudo-random generation algorithm (PRGA)**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=4)]

[](https://en.wikipedia.org/wiki/File:RC4.svg)

The lookup stage of RC4. The output byte is selected by looking up the values of S[i] and S[j], adding them together modulo 256, and then using the sum as an index into S; S(S[i] + S[j]) is used as a byte of the key stream, K.

For as many iterations as are needed, the PRGA modifies the state and outputs a byte of the keystream. In each iteration, the PRGA:

* increments *i*
* looks up the *i*th element of S, S[*i*], and adds that to *j*
* exchanges the values of S[*i*] and S[*j*] then uses the sum S[*i*] + S[*j*] (modulo 256) as an index to fetch a third element of S (the keystream value K below)
* then bitwise exclusive ORed ([XORed](https://en.wikipedia.org/wiki/Exclusive_or)) with the next byte of the message to produce the next byte of either ciphertext or plaintext.

Each element of S is swapped with another element at least once every 256 iterations.

i := 0

j := 0

**while** GeneratingOutput:

i := (i + 1) mod 256

j := (j + S[i]) mod 256

[swap values](https://en.wikipedia.org/wiki/Swap_(computer_science)) of S[i] and S[j]

K := S[(S[i] + S[j]) mod 256]

output K

**endwhile**

**RC4-based random number generators**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=5)]

Several [operating systems](https://en.wikipedia.org/wiki/Operating_system) include arc4random, an API originating in [OpenBSD](https://en.wikipedia.org/wiki/OpenBSD_security_features) providing access to a random number generator originally based on RC4. In OpenBSD 5.5, released in May 2014, arc4random was modified to use [ChaCha20](https://en.wikipedia.org/wiki/ChaCha20).[[15]](https://en.wikipedia.org/wiki/RC4#cite_note-15)[[16]](https://en.wikipedia.org/wiki/RC4#cite_note-16) The implementations of arc4random in [FreeBSD](https://en.wikipedia.org/wiki/FreeBSD), [NetBSD](https://en.wikipedia.org/wiki/NetBSD)[[17]](https://en.wikipedia.org/wiki/RC4#cite_note-17)[[18]](https://en.wikipedia.org/wiki/RC4#cite_note-18) and [Linux](https://en.wikipedia.org/wiki/Linux)'s libbsd[[19]](https://en.wikipedia.org/wiki/RC4" \l "cite_note-19) also use ChaCha20. According to manual pages shipped with the operating system, in the 2017 release of its [desktop](https://en.wikipedia.org/wiki/MacOS) and [mobile](https://en.wikipedia.org/wiki/IOS) operating systems, Apple replaced RC4 with AES in its implementation of arc4random. [Man pages](https://en.wikipedia.org/wiki/Man_page) for the new arc4random include the [backronym](https://en.wikipedia.org/wiki/Backronym) "A Replacement Call for Random" for ARC4 as a mnemonic,[[20]](https://en.wikipedia.org/wiki/RC4#cite_note-arc4random-obsd-20) as it provides better random data than [rand()](https://en.wikipedia.org/wiki/Rand()) does.

Proposed new random number generators are often compared to the RC4 random number generator.[[21]](https://en.wikipedia.org/wiki/RC4#cite_note-21)[[22]](https://en.wikipedia.org/wiki/RC4#cite_note-22)

Several attacks on RC4 are able to [distinguish its output from a random sequence](https://en.wikipedia.org/wiki/Ciphertext_indistinguishability#Indistinguishable_from_random_noise).[[23]](https://en.wikipedia.org/wiki/RC4#cite_note-mantin-23)

**Implementation**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=6)]

Many stream ciphers are based on [linear-feedback shift registers](https://en.wikipedia.org/wiki/Linear-feedback_shift_register) (LFSRs), which, while efficient in hardware, are less so in software. The design of RC4 avoids the use of LFSRs and is ideal for software implementation, as it requires only byte manipulations. It uses 256 bytes of memory for the state array, S[0] through S[255], k bytes of memory for the key, key[0] through key[k-1], and integer variables, i, j, and K. Performing a modular reduction of some value modulo 256 can be done with a [bitwise AND](https://en.wikipedia.org/wiki/Bitwise_AND) with 255 (which is equivalent to taking the low-order byte of the value in question).

**Test vectors**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=7)]

These test vectors are not official, but convenient for anyone testing their own RC4 program. The keys and plaintext are [ASCII](https://en.wikipedia.org/wiki/ASCII), the keystream and ciphertext are in [hexadecimal](https://en.wikipedia.org/wiki/Hexadecimal).

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **Keystream** | **Plaintext** | **Ciphertext** |
| Key | EB9F7781B734CA72A719… | Plaintext | BBF316E8D940AF0AD3 |
| Wiki | 6044DB6D41B7… | pedia | 1021BF0420 |
| Secret | 04D46B053CA87B59… | Attack at dawn | 45A01F645FC35B383552544B9BF5 |

Security[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=8)]

Unlike a modern stream cipher (such as those in [eSTREAM](https://en.wikipedia.org/wiki/ESTREAM" \o "ESTREAM)), RC4 does not take a separate [nonce](https://en.wikipedia.org/wiki/Cryptographic_nonce) alongside the key. This means that if a single long-term key is to be used to securely encrypt multiple streams, the protocol must specify how to combine the nonce and the long-term key to generate the stream key for RC4. One approach to addressing this is to generate a "fresh" RC4 key by [hashing](https://en.wikipedia.org/wiki/Cryptographic_hash_function) a long-term key with a [nonce](https://en.wikipedia.org/wiki/Cryptographic_nonce). However, many applications that use RC4 simply concatenate key and nonce; RC4's weak [key schedule](https://en.wikipedia.org/wiki/Key_schedule) then gives rise to [related key attacks](https://en.wikipedia.org/wiki/Related_key_attack), like the [Fluhrer, Mantin and Shamir attack](https://en.wikipedia.org/wiki/Fluhrer,_Mantin_and_Shamir_attack) (which is famous for breaking the [WEP](https://en.wikipedia.org/wiki/Wired_Equivalent_Privacy) standard).[[24]](https://en.wikipedia.org/wiki/RC4#cite_note-24)

Because RC4 is a [stream cipher](https://en.wikipedia.org/wiki/Stream_cipher), it is more [malleable](https://en.wikipedia.org/wiki/Malleability_(cryptography)) than common [block ciphers](https://en.wikipedia.org/wiki/Block_cipher). If not used together with a strong [message authentication code](https://en.wikipedia.org/wiki/Message_authentication_code) (MAC), then encryption is vulnerable to a [bit-flipping attack](https://en.wikipedia.org/wiki/Bit-flipping_attack). The cipher is also vulnerable to a [stream cipher attack](https://en.wikipedia.org/wiki/Stream_cipher_attack) if not implemented correctly.[[25]](https://en.wikipedia.org/wiki/RC4#cite_note-25)

It is noteworthy, however, that RC4, being a stream cipher, was for a period of time the only common cipher that was immune[[26]](https://en.wikipedia.org/wiki/RC4#cite_note-26) to the 2011 [BEAST attack](https://en.wikipedia.org/wiki/BEAST_attack) on [TLS 1.0](https://en.wikipedia.org/wiki/Transport_Layer_Security#TLS_1.0_.28SSL_3.1.29). The attack exploits a known weakness in the way [cipher block chaining mode](https://en.wikipedia.org/wiki/Block_cipher_modes_of_operation#Cipher-block_chaining_.28CBC.29) is used with all of the other ciphers supported by TLS 1.0, which are all block ciphers.

In March 2013, there were new attack scenarios proposed by Isobe, Ohigashi, Watanabe and Morii,[[27]](https://en.wikipedia.org/wiki/RC4#cite_note-27) as well as AlFardan, Bernstein, Paterson, Poettering and Schuldt that use new statistical biases in RC4 key table[[28]](https://en.wikipedia.org/wiki/RC4#cite_note-28) to recover plaintext with large number of TLS encryptions.[[29]](https://en.wikipedia.org/wiki/RC4#cite_note-29)[[30]](https://en.wikipedia.org/wiki/RC4#cite_note-30)

The use of RC4 in TLS is prohibited by [RFC 7465](https://tools.ietf.org/html/rfc7465) published in February 2015.

**Roos's biases and key reconstruction from permutation**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=9)]

In 1995, Andrew Roos experimentally observed that the first byte of the keystream is correlated to the first three bytes of the key and the first few bytes of the permutation after the KSA are correlated to some linear combination of the key bytes.[[31]](https://en.wikipedia.org/wiki/RC4#cite_note-31) These biases remained unexplained until 2007, when Goutam Paul, Siddheshwar Rathi and Subhamoy Maitra[[32]](https://en.wikipedia.org/wiki/RC4#cite_note-32) proved the keystream–key correlation and in another work Goutam Paul and Subhamoy Maitra[[33]](https://en.wikipedia.org/wiki/RC4#cite_note-33) proved the permutation–key correlations. The latter work also used the permutation–key correlations to design the first algorithm for complete key reconstruction from the final permutation after the KSA, without any assumption on the key or [initialization vector](https://en.wikipedia.org/wiki/Initialization_vector). This algorithm has a constant probability of success in a time which is the square root of the exhaustive key search complexity. Subsequently, many other works have been performed on key reconstruction from RC4 internal states.[[34]](https://en.wikipedia.org/wiki/RC4#cite_note-34)[[35]](https://en.wikipedia.org/wiki/RC4#cite_note-35)[[36]](https://en.wikipedia.org/wiki/RC4#cite_note-36) Subhamoy Maitra and Goutam Paul[[37]](https://en.wikipedia.org/wiki/RC4#cite_note-37) also showed that the Roos-type biases still persist even when one considers nested permutation indices, like S[S[i]] or S[S[S[i]]]. These types of biases are used in some of the later key reconstruction methods for increasing the success probability.

**Biased outputs of the RC4**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=10)]

The keystream generated by the RC4 is biased in varying degrees towards certain sequences making it vulnerable to [distinguishing attacks](https://en.wikipedia.org/wiki/Distinguishing_attack). The best such attack is due to Itsik Mantin and [Adi Shamir](https://en.wikipedia.org/wiki/Adi_Shamir) who showed that the second output byte of the cipher was biased toward zero with probability 1/128 (instead of 1/256). This is due to the fact that if the third byte of the original state is zero, and the second byte is not equal to 2, then the second output byte is always zero. Such bias can be detected by observing only 256 bytes.[[23]](https://en.wikipedia.org/wiki/RC4#cite_note-mantin-23)

[Souradyuti Paul](https://en.wikipedia.org/wiki/Souradyuti_Paul) and [Bart Preneel](https://en.wikipedia.org/wiki/Bart_Preneel) of [COSIC](https://en.wikipedia.org/wiki/COSIC) showed that the first and the second bytes of the RC4 were also biased. The number of required samples to detect this bias is 225 bytes.[[38]](https://en.wikipedia.org/wiki/RC4#cite_note-38)

[Scott Fluhrer](https://en.wikipedia.org/w/index.php?title=Scott_Fluhrer&action=edit&redlink=1) and David McGrew also showed such attacks which distinguished the keystream of the RC4 from a random stream given a gigabyte of output.[[39]](https://en.wikipedia.org/wiki/RC4#cite_note-39)

The complete characterization of a single step of RC4 PRGA was performed by Riddhipratim Basu, Shirshendu Ganguly, Subhamoy Maitra, and Goutam Paul.[[40]](https://en.wikipedia.org/wiki/RC4#cite_note-40) Considering all the permutations, they prove that the distribution of the output is not uniform given i and j, and as a consequence, information about j is always leaked into the output.

**Fluhrer, Mantin and Shamir attack**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=11)]

*Main article:*[*Fluhrer, Mantin and Shamir attack*](https://en.wikipedia.org/wiki/Fluhrer,_Mantin_and_Shamir_attack)

In 2001, a new and surprising discovery was made by [Fluhrer](https://en.wikipedia.org/w/index.php?title=Scott_Fluhrer&action=edit&redlink=1), [Mantin](https://en.wikipedia.org/w/index.php?title=Itsik_Mantin&action=edit&redlink=1" \o "Itsik Mantin (page does not exist)) and [Shamir](https://en.wikipedia.org/wiki/Adi_Shamir): over all the possible RC4 keys, the statistics for the first few bytes of output keystream are strongly non-random, leaking information about the key. If the nonce and long-term key are simply concatenated to generate the RC4 key, this long-term key can be discovered by analysing a large number of messages encrypted with this key.[[41]](https://en.wikipedia.org/wiki/RC4#cite_note-41) This and related effects were then used to break the [WEP](https://en.wikipedia.org/wiki/Wired_Equivalent_Privacy) ("wired equivalent privacy") encryption used with [802.11](https://en.wikipedia.org/wiki/802.11) [wireless networks](https://en.wikipedia.org/wiki/Wireless_network). This caused a scramble for a standards-based replacement for WEP in the 802.11 market, and led to the [IEEE 802.11i](https://en.wikipedia.org/wiki/IEEE_802.11i) effort and [WPA](https://en.wikipedia.org/wiki/Wi-Fi_Protected_Access).[[42]](https://en.wikipedia.org/wiki/RC4#cite_note-42)

Protocols can defend against this attack by discarding the initial portion of the keystream. Such a modified algorithm is traditionally called "RC4-drop[*n*]", where *n* is the number of initial keystream bytes that are dropped. The SCAN default is *n* = 768 bytes, but a conservative value would be *n* = 3072 bytes.[[43]](https://en.wikipedia.org/wiki/RC4#cite_note-43)

The Fluhrer, Mantin and Shamir attack does not apply to RC4-based SSL, since SSL generates the encryption keys it uses for RC4 by hashing, meaning that different SSL sessions have unrelated keys.[[44]](https://en.wikipedia.org/wiki/RC4#cite_note-44)

**Klein's attack**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=12)]

In 2005, Andreas Klein presented an analysis of the RC4 stream cipher showing more correlations between the RC4 keystream and the key.[[45]](https://en.wikipedia.org/wiki/RC4#cite_note-45) [Erik Tews](https://en.wikipedia.org/w/index.php?title=Erik_Tews&action=edit&redlink=1), [Ralf-Philipp Weinmann](https://en.wikipedia.org/w/index.php?title=Ralf-Philipp_Weinmann&action=edit&redlink=1), and [Andrei Pychkine](https://en.wikipedia.org/w/index.php?title=Andrei_Pychkine&action=edit&redlink=1) used this analysis to create aircrack-ptw, a tool which cracks 104-bit RC4 used in 128-bit WEP in under a minute.[[46]](https://en.wikipedia.org/wiki/RC4#cite_note-46) Whereas the Fluhrer, Mantin, and Shamir attack used around 10 million messages, aircrack-ptw can break 104-bit keys in 40,000 frames with 50% probability, or in 85,000 frames with 95% probability.

**Combinatorial problem**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=13)]

A combinatorial problem related to the number of inputs and outputs of the RC4 cipher was first posed by [Itsik Mantin](https://en.wikipedia.org/w/index.php?title=Itsik_Mantin&action=edit&redlink=1" \o "Itsik Mantin (page does not exist)) and [Adi Shamir](https://en.wikipedia.org/wiki/Adi_Shamir) in 2001, whereby, of the total 256 elements in the typical state of RC4, if *x* number of elements (*x* ≤ 256) are *only* known (all other elements can be assumed empty), then the maximum number of elements that can be produced deterministically is also *x* in the next 256 rounds. This conjecture was put to rest in 2004 with a formal proof given by [Souradyuti Paul](https://en.wikipedia.org/wiki/Souradyuti_Paul" \o "Souradyuti Paul) and [Bart Preneel](https://en.wikipedia.org/wiki/Bart_Preneel).[[47]](https://en.wikipedia.org/wiki/RC4#cite_note-47)

**Royal Holloway attack**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=14)]

In 2013, a group of security researchers at the Information Security Group at Royal Holloway, University of London reported an attack that can become effective using only 234 encrypted messages.[[48]](https://en.wikipedia.org/wiki/RC4#cite_note-48)[[49]](https://en.wikipedia.org/wiki/RC4#cite_note-49)[[50]](https://en.wikipedia.org/wiki/RC4#cite_note-50) While yet not a practical attack for most purposes, this result is sufficiently close to one that it has led to speculation that it is plausible that some state cryptologic agencies may already have better attacks that render RC4 insecure.[[6]](https://en.wikipedia.org/wiki/RC4#cite_note-Leyden20130906-6) Given that, as of 2013, a large amount of [TLS](https://en.wikipedia.org/wiki/Transport_Layer_Security) traffic uses RC4 to avoid attacks on block ciphers that use [cipher block chaining](https://en.wikipedia.org/wiki/Cipher_block_chaining), if these hypothetical better attacks exist, then this would make the TLS-with-RC4 combination insecure against such attackers in a large number of practical scenarios.[[6]](https://en.wikipedia.org/wiki/RC4#cite_note-Leyden20130906-6)

In March 2015 researcher to Royal Holloway announced improvements to their attack, providing a 226 attack against passwords encrypted with RC4, as used in TLS.[[51]](https://en.wikipedia.org/wiki/RC4#cite_note-51)

**Bar-mitzvah attack**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=15)]

*Main article:*[*Bar-mitzvah attack*](https://en.wikipedia.org/wiki/Bar-mitzvah_attack)

On the Black Hat Asia 2015, Itsik Mantin presented another attack against SSL using RC4 cipher.[[52]](https://en.wikipedia.org/wiki/RC4#cite_note-52)[[53]](https://en.wikipedia.org/wiki/RC4#cite_note-53)

**NOMORE attack**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=16)]

In 2015, security researchers from [KU Leuven](https://en.wikipedia.org/wiki/Katholieke_Universiteit_Leuven) presented new attacks against RC4 in both [TLS](https://en.wikipedia.org/wiki/Transport_Layer_Security) and [WPA-TKIP](https://en.wikipedia.org/wiki/Temporal_Key_Integrity_Protocol).[[54]](https://en.wikipedia.org/wiki/RC4#cite_note-rc4nomore-54) Dubbed the Numerous Occurrence MOnitoring & Recovery Exploit (NOMORE) attack, it is the first attack of its kind that was demonstrated in practice. Their attack against [TLS](https://en.wikipedia.org/wiki/Transport_Layer_Security) can decrypt a secure [HTTP cookie](https://en.wikipedia.org/wiki/HTTP_cookie) within 75 hours. The attack against WPA-TKIP can be completed within an hour, and allows an attacker to decrypt and inject arbitrary packets.

RC4 variants[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=17)]

As mentioned above, the most important weakness of RC4 comes from the insufficient key schedule; the first bytes of output reveal information about the key. This can be corrected by simply discarding some initial portion of the output stream.[[55]](https://en.wikipedia.org/wiki/RC4#cite_note-55) This is known as RC4-drop*N*, where *N* is typically a multiple of 256, such as 768 or 1024.

A number of attempts have been made to strengthen RC4, notably Spritz, RC4A, [VMPC](https://en.wikipedia.org/wiki/Variably_Modified_Permutation_Composition), and RC4+.

**RC4A**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=18)]

[Souradyuti Paul](https://en.wikipedia.org/wiki/Souradyuti_Paul) and [Bart Preneel](https://en.wikipedia.org/wiki/Bart_Preneel) have proposed an RC4 variant, which they call RC4A.[[56]](https://en.wikipedia.org/wiki/RC4#cite_note-56)

RC4A uses two state arrays S1 and S2, and two indexes *j1* and *j2*. Each time *i* is incremented, two bytes are generated:

1. First, the basic RC4 algorithm is performed using S1 and *j1*, but in the last step, S1[*i*]+S1[*j1*] is looked up in S2.
2. Second, the operation is repeated (without incrementing *i* again) on S2 and *j2*, and S1[S2[*i*]+S2[*j2*]] is output.

Thus, the algorithm is:

*All arithmetic is performed modulo 256*

i := 0

j1 := 0

j2 := 0

**while** GeneratingOutput:

i := i + 1

j1 := j1 + S1[i]

[swap values](https://en.wikipedia.org/wiki/Swap_(computer_science)) of S1[i] and S1[j1]

**output** S2[S1[i] + S1[j1]]

j2 := j2 + S2[i]

swap values of S2[i] and S2[j2]

**output** S1[S2[i] + S2[j2]]

**endwhile**

Although the algorithm required the same number of operations per output byte, there is greater parallelism than RC4, providing a possible speed improvement.

Although stronger than RC4, this algorithm has also been attacked, with Alexander Maximov[[57]](https://en.wikipedia.org/wiki/RC4" \l "cite_note-57) and a team from NEC[[58]](https://en.wikipedia.org/wiki/RC4#cite_note-nec-58) developing ways to distinguish its output from a truly random sequence.

**VMPC**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=19)]

*Main article:*[*Variably Modified Permutation Composition*](https://en.wikipedia.org/wiki/Variably_Modified_Permutation_Composition)

Variably Modified Permutation Composition (VMPC) is another RC4 variant.[[59]](https://en.wikipedia.org/wiki/RC4#cite_note-59) It uses similar key schedule as RC4, with j := S[(j + S[i] + key[i mod keylength]) mod 256] iterating 3 × 256 = 768 times rather than 256, and with an optional additional 768 iterations to incorporate an initial vector. The output generation function operates as follows:

*All arithmetic is performed modulo 256.*

i := 0

**while** GeneratingOutput:

a := S[i]

j := S[j + a]

**output** S[S[S[j] + 1]]

Swap S[i] and S[j] (*b := S[j]; S[i] := b; S[j] := a)*)

i := i + 1

**endwhile**

This was attacked in the same papers as RC4A, and can be distinguished within 238 output bytes.[[60]](https://en.wikipedia.org/wiki/RC4#cite_note-maximov-60)[[58]](https://en.wikipedia.org/wiki/RC4#cite_note-nec-58)

**RC4+**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=20)]

RC4+ is a modified version of RC4 with a more complex three-phase key schedule (taking about three times as long as RC4, or the same as RC4-drop512), and a more complex output function which performs four additional lookups in the S array for each byte output, taking approximately 1.7 times as long as basic RC4.[[61]](https://en.wikipedia.org/wiki/RC4#cite_note-rc4+-61)

*All arithmetic modulo 256.* << *and* >> *are left and right shift,* ⊕ *is exclusive OR*

**while** GeneratingOutput:

i := i + 1

a := S[i]

j := j + a

Swap S[i] and S[j] (*b := S[j]; S[i] := b; S[j] := a*)

c := S[i<<5 ⊕ j>>3] + S[j<<5 ⊕ i>>3]

**output** (S[a+b] + S[c⊕0xAA]) ⊕ S[j+b]

**endwhile**

This algorithm has not been analyzed significantly.

**Spritz**[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=21)]

In 2014, Ronald Rivest gave a talk and co-wrote a paper[[14]](https://en.wikipedia.org/wiki/RC4#cite_note-Rivest2014-14) on an updated redesign called [Spritz](https://en.wikipedia.org/wiki/RC4#Spritz). A hardware accelerator of Spritz was published in Secrypt, 2016[[62]](https://en.wikipedia.org/wiki/RC4#cite_note-62) and shows that due to multiple nested calls required to produce output bytes, Spritz performs rather slowly compared to other hash functions such as SHA-3 and the best known hardware implementation of RC4.

The algorithm is:[[14]](https://en.wikipedia.org/wiki/RC4#cite_note-Rivest2014-14)

*All arithmetic is performed modulo 256*

**while** GeneratingOutput:

i := i + w

j := k + S[j + S[i]]

k := k + i + S[j]

swap values of S[i] and S[j]

**output** z := S[j + S[i + S[z + k]]]

**endwhile**

The value *w*, is [relatively prime](https://en.wikipedia.org/wiki/Relatively_prime) to the size of the S array. So after 256 iterations of this inner loop, the value *i* (incremented by *w* every iteration) has taken on all possible values 0...255, and every byte in the S array has been swapped at least once.

Like other [sponge functions](https://en.wikipedia.org/wiki/Sponge_function), Spritz can be used to build a cryptographic hash function, a deterministic random bit generator ([DRBG](https://en.wikipedia.org/wiki/DRBG)), an encryption algorithm that supports [authenticated encryption](https://en.wikipedia.org/wiki/Authenticated_encryption) with associated data (AEAD), etc.[[14]](https://en.wikipedia.org/wiki/RC4#cite_note-Rivest2014-14)

In 2016, Banik and Isobe proposed an attack that can distinguish Spritz from random noise.[[63]](https://en.wikipedia.org/wiki/RC4#cite_note-63)

RC4-based protocols[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=22)]

* [WEP](https://en.wikipedia.org/wiki/Wired_Equivalent_Privacy)
* [WPA](https://en.wikipedia.org/wiki/Wi-Fi_Protected_Access) (default algorithm, but can be configured to use [AES-CCMP](https://en.wikipedia.org/wiki/AES-CCMP) instead of RC4)
* [BitTorrent protocol encryption](https://en.wikipedia.org/wiki/BitTorrent_protocol_encryption)
* [Microsoft Office XP](https://en.wikipedia.org/wiki/Microsoft_Office_XP) (insecure implementation since nonce remains unchanged when documents get modified[[64]](https://en.wikipedia.org/wiki/RC4#cite_note-64))
* [Microsoft Point-to-Point Encryption](https://en.wikipedia.org/wiki/Microsoft_Point-to-Point_Encryption)
* [Transport Layer Security](https://en.wikipedia.org/wiki/Transport_Layer_Security) / [Secure Sockets Layer](https://en.wikipedia.org/wiki/Secure_Sockets_Layer) (was optional and then the use of RC4 was prohibited in [RFC 7465](https://tools.ietf.org/html/rfc7465))
* [Secure Shell](https://en.wikipedia.org/wiki/Secure_Shell) (optionally)
* [Remote Desktop Protocol](https://en.wikipedia.org/wiki/Remote_Desktop_Protocol) (optionally)
* [Kerberos](https://en.wikipedia.org/wiki/Kerberos_(protocol)) (optionally)
* [SASL](https://en.wikipedia.org/wiki/Simple_Authentication_and_Security_Layer) Mechanism Digest-MD5 (optionally, *historic*, obsoleted in [RFC 6331](https://tools.ietf.org/html/rfc6331))
* [Gpcode.AK](https://en.wikipedia.org/w/index.php?title=Gpcode.AK&action=edit&redlink=1), an early June 2008 computer virus for Microsoft Windows, which takes documents hostage for [ransom](https://en.wikipedia.org/wiki/Ransom) by obscuring them with RC4 and RSA-1024 encryption
* [PDF](https://en.wikipedia.org/wiki/Portable_Document_Format)
* [Skype](https://en.wikipedia.org/wiki/Skype) (in modified form)[[65]](https://en.wikipedia.org/wiki/RC4#cite_note-65)

Where a protocol is marked with "(optionally)", RC4 is one of multiple ciphers the system can be configured to use.

See also[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=23)]

* [TEA](https://en.wikipedia.org/wiki/Tiny_Encryption_Algorithm), [Block TEA](https://en.wikipedia.org/wiki/XTEA) also known as [eXtended TEA](https://en.wikipedia.org/wiki/XTEA" \o "XTEA) and [Corrected Block TEA](https://en.wikipedia.org/wiki/XXTEA) – A family of [block ciphers](https://en.wikipedia.org/wiki/Block_cipher) that, like RC4, are designed to be very simple to implement.
* [Advanced Encryption Standard](https://en.wikipedia.org/wiki/Advanced_Encryption_Standard)
* [CipherSaber](https://en.wikipedia.org/wiki/CipherSaber)

References[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=24)]

* 1. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-1) *P. Prasithsangaree & P. Krishnamurthy (2003).*[*"Analysis of Energy Consumption of RC4 and AES Algorithms in Wireless LANs"*](https://web.archive.org/web/20131203082918/http:/www.sis.pitt.edu/~is3966/group5_paper2.pdf)*(PDF). Archived from*[*the original*](http://www.sis.pitt.edu/~is3966/group5_paper2.pdf)*(PDF) on 3 December 2013.*
  2. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-2) [*"Crypto++ 5.6.0 Benchmarks"*](http://www.cryptopp.com/benchmarks.html)*. Retrieved 22 September 2015.*
  3. ^ [Jump up to:***a***](https://en.wikipedia.org/wiki/RC4#cite_ref-rfc7465_3-0) [***b***](https://en.wikipedia.org/wiki/RC4#cite_ref-rfc7465_3-1) *Andrei Popov (February 2015).*[*Prohibiting RC4 Cipher Suites*](https://tools.ietf.org/html/rfc7465)*.*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.17487/RFC7465*](https://doi.org/10.17487%2FRFC7465)*.*[*RFC*](https://en.wikipedia.org/wiki/RFC_(identifier))[*7465*](https://tools.ietf.org/html/rfc7465)*.*
  4. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-4) *Lucian Constantin (14 May 2014).*[*"Microsoft continues RC4 encryption phase-out plan with .NET security updates"*](http://www.computerworld.com/article/2489395/encryption/microsoft-continues-rc4-encryption-phase-out-plan-with--net-security-updates.html)*. ComputerWorld.*
  5. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-5) *J. Katz; Y. Lindell (2014), Introduction to Modern Cryptography, Chapman and Hall/CRC, p. 77*
  6. ^ [Jump up to:***a***](https://en.wikipedia.org/wiki/RC4#cite_ref-Leyden20130906_6-0) [***b***](https://en.wikipedia.org/wiki/RC4#cite_ref-Leyden20130906_6-1) [***c***](https://en.wikipedia.org/wiki/RC4#cite_ref-Leyden20130906_6-2) *John Leyden (6 September 2013).*[*"That earth-shattering NSA crypto-cracking: Have spooks smashed RC4?"*](https://www.theregister.co.uk/2013/09/06/nsa_cryptobreaking_bullrun_analysis/)*. The Register.*
  7. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-7) [*"Mozilla Security Server Side TLS Recommended Configurations"*](https://wiki.mozilla.org/Security/Server_Side_TLS)*. Mozilla. Retrieved 3 January 2015.*
  8. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-8) [*"Security Advisory 2868725: Recommendation to disable RC4"*](http://blogs.technet.com/b/srd/archive/2013/11/12/security-advisory-2868725-recommendation-to-disable-rc4.aspx)*. Microsoft. 12 November 2013. Retrieved 4 December 2013.*
  9. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-9) [Rivest FAQ](http://people.csail.mit.edu/rivest/faq.html" \l "Ron)
  10. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-10) [*"Thank you Bob Anderson"*](https://web.archive.org/web/20010722163902/http:/cypherpunks.venona.com/date/1994/09/msg00304.html)*. [Cypherpunks](https://en.wikipedia.org/wiki/Cypherpunk" \o "Cypherpunk) (Mailing list). 9 September 1994. Archived from*[*the original*](http://cypherpunks.venona.com/date/1994/09/msg00304.html)*on 22 July 2001. Retrieved 28 May 2007.*
  11. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-11) *Bob Jenkins (15 September 1994).*[*"Re: RC4 ?"*](https://groups.google.com/d/msg/sci.crypt/JsO3xEATGFA/-wO4ttv7BCYJ)*.*[*Newsgroup*](https://en.wikipedia.org/wiki/Usenet_newsgroup)*:*[*sci.crypt*](news:sci.crypt)*.*[*Usenet:*](https://en.wikipedia.org/wiki/Usenet_(identifier))[*359qjg$55v$1@mhadg.production.compuserve.com*](news:359qjg$55v$1@mhadg.production.compuserve.com)*.*
  12. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-12) [*"Manual Pages: arc4random"*](https://man.openbsd.org/OpenBSD-5.4/arc4random.3)*. 5 June 2013. Retrieved 2 February 2018.*
  13. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-13) [6.857 Computer and Network Security Spring 2008: Lectures and Handouts](https://courses.csail.mit.edu/6.857/2008/lecture.html)
  14. ^ [Jump up to:***a***](https://en.wikipedia.org/wiki/RC4#cite_ref-Rivest2014_14-0) [***b***](https://en.wikipedia.org/wiki/RC4#cite_ref-Rivest2014_14-1) [***c***](https://en.wikipedia.org/wiki/RC4#cite_ref-Rivest2014_14-2) [***d***](https://en.wikipedia.org/wiki/RC4#cite_ref-Rivest2014_14-3) *Rivest, Ron; Schuldt, Jacob (27 October 2014).*[*"Spritz – a spongy RC4-like stream cipher and hash function"*](http://people.csail.mit.edu/rivest/pubs/RS14.pdf)*(PDF). Retrieved 26 October 2014.*
  15. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-15) [*"OpenBSD 5.5"*](http://www.openbsd.org/55.html)*. Retrieved 21 September 2014.*
  16. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-16) *[deraadt](https://en.wikipedia.org/wiki/Theo_de_Raadt" \o "Theo de Raadt), ed. (21 July 2014).*[*"libc/crypt/arc4random.c"*](http://bxr.su/OpenBSD/lib/libc/crypt/arc4random.c)*. BSD Cross Reference, OpenBSD src/lib/. Retrieved 13 January 2015. ChaCha based random number generator for OpenBSD.*
  17. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-17) *riastradh, ed. (16 November 2014).*[*"libc/gen/arc4random.c"*](http://bxr.su/NetBSD/lib/libc/gen/arc4random.c)*. BSD Cross Reference, NetBSD src/lib/. Retrieved 13 January 2015. Legacy arc4random(3) API from OpenBSD reimplemented using the ChaCha20 PRF, with per-thread state.*
  18. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-18) [*"arc4random – NetBSD Manual Pages"*](http://netbsd.gw.com/cgi-bin/man-cgi?arc4random++NetBSD-current)*. Retrieved 6 January2015.*
  19. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-19) [*"Update arc4random module from OpenBSD and LibreSSL"*](https://cgit.freedesktop.org/libbsd/commit/?id=874a0e5)*. Retrieved 6 January 2016.*
  20. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-arc4random-obsd_20-0) [*"arc4random(3)"*](http://man.openbsd.org/arc4random.3)*. OpenBSD.*
  21. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-21) Bartosz Zoltak. ["VMPC-R: Cryptographically Secure Pseudo-Random Number Generator, Alternative to RC4".](https://eprint.iacr.org/2013/768.pdf) 2010?
  22. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-22) Chefranov, A.G. ["Pseudo-Random Number Generator RC4 Period Improvement"](http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=4022919&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D4022919). 2006.
  23. ^ [Jump up to:***a***](https://en.wikipedia.org/wiki/RC4#cite_ref-mantin_23-0) [***b***](https://en.wikipedia.org/wiki/RC4#cite_ref-mantin_23-1) *Itsik Mantin,*[*Adi Shamir*](https://en.wikipedia.org/wiki/Adi_Shamir)*(2001).*[*"A Practical Attack on Broadcast RC4"*](http://saluc.engr.uconn.edu/refs/stream_cipher/mantin01attackRC4.pdf)*(PDF): 152–164.*
  24. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-24) [*"RSA Security Response to Weaknesses in Key Scheduling Algorithm of RC4"*](http://www.emc.com/emc-plus/rsa-labs/historical/rsa-security-response-weaknesses-algorithm-rc4.htm)*. RSA Laboratories. 1 September 2001.*
  25. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-25) *Sklyarov, Dmitry (2004).*[*Hidden Keys to Software Break-Ins and Unauthorized Entry*](https://books.google.com/books?id=dQE6LGep6wEC)*. A-List Publishing. pp. 92–93.*[*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier))[*978-1931769303*](https://en.wikipedia.org/wiki/Special:BookSources/978-1931769303)*.*
  26. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-26) ["ssl - Safest ciphers to use with the BEAST? (TLS 1.0 exploit) I've read that RC4 is immune - Server Fault"](http://serverfault.com/questions/315042/). *serverfault.com*.
  27. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-27) *Isobe, Takanori; Ohigashi, Toshihiro (10–13 March 2013).*[*"Security of RC4 Stream Cipher"*](http://home.hiroshima-u.ac.jp/ohigashi/rc4/)*. Hiroshima University. Retrieved 27 October 2014.*
  28. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-28) *Pouyan Sepehrdad; Serge Vaudenay; Martin Vuagnoux (2011). Discovery and Exploitation of New Biases in RC4. Lecture Notes in Computer Science.****6544****. pp. 74–91.*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.1007/978-3-642-19574-7\_5*](https://doi.org/10.1007%2F978-3-642-19574-7_5)*.*[*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier))[*978-3-642-19573-0*](https://en.wikipedia.org/wiki/Special:BookSources/978-3-642-19573-0)*.*
  29. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-29) *Green, Matthew (12 March 2013).*[*"Attack of the week: RC4 is kind of broken in TLS"*](http://blog.cryptographyengineering.com/2013/03/attack-of-week-rc4-is-kind-of-broken-in.html)*. Cryptography Engineering. Retrieved 12 March2013.*
  30. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-30) *Nadhem AlFardan; Dan Bernstein; Kenny Paterson; Bertram Poettering; Jacob Schuldt.*[*"On the Security of RC4 in TLS"*](http://www.isg.rhul.ac.uk/tls/)*. Royal Holloway University of London. Retrieved 13 March 2013.*
  31. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-31) Andrew Roos. A Class of Weak Keys in the RC4 Stream Cipher. Two posts in sci.crypt, message-id 43u1eh$1j3@hermes.is.co.za and 44ebge$llf@hermes.is.co.za, 1995.
  32. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-32) Goutam Paul, Siddheshwar Rathi and Subhamoy Maitra. On Non-negligible Bias of the First Output Byte of RC4 towards the First Three Bytes of the Secret Key. Proceedings of the International Workshop on Coding and Cryptography (WCC) 2007, pages 285–294 and Designs, Codes and Cryptography Journal, pages 123–134, vol. 49, no. 1-3, December 2008.
  33. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-33) Goutam Paul and Subhamoy Maitra. Permutation after RC4 Key Scheduling Reveals the Secret Key. SAC 2007, pages 360–377, vol. 4876, [Lecture Notes in Computer Science](https://en.wikipedia.org/wiki/Lecture_Notes_in_Computer_Science), Springer.
  34. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-34) Eli Biham and Yaniv Carmeli. Efficient Reconstruction of RC4 Keys from Internal States. FSE 2008, pages 270–288, vol. 5086, Lecture Notes in Computer Science, Springer.
  35. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-35) Mete Akgun, Pinar Kavak, Huseyin Demirci. New Results on the Key Scheduling Algorithm of RC4. INDOCRYPT 2008, pages 40–52, vol. 5365, Lecture Notes in Computer Science, Springer.
  36. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-36) Riddhipratim Basu, Subhamoy Maitra, Goutam Paul and Tanmoy Talukdar. On Some Sequences of the Secret Pseudo-random Index j in RC4 Key Scheduling. Proceedings of the 18th International Symposium on Applied Algebra, Algebraic Algorithms and Error Correcting Codes (AAECC), 8–12 June 2009, Tarragona, Spain, pages 137–148, vol. 5527, Lecture Notes in Computer Science, Springer.
  37. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-37) Subhamoy Maitra and Goutam Paul. New Form of Permutation Bias and Secret Key Leakage in Keystream Bytes of RC4. Proceedings of the 15th Fast Software Encryption (FSE) Workshop, 10–13 February 2008, Lausanne, Switzerland, pages 253–269, vol. 5086, Lecture Notes in Computer Science, Springer.
  38. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-38) *[Souradyuti Paul](https://en.wikipedia.org/wiki/Souradyuti_Paul" \o "Souradyuti Paul),*[*Bart Preneel*](https://en.wikipedia.org/wiki/Bart_Preneel)*.*[*"Analysis of Non-fortuitous Predictive States of the RC4 Keystream Generator"*](http://www.cosic.esat.kuleuven.be/publications/article-86.pdf)*(PDF): 52–67.*
  39. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-39) *Scott R. Fluhrer, David A. McGrew.*[*"Statistical Analysis of the Alleged RC4 Keystream Generator"*](https://web.archive.org/web/20140502020708/http:/www.wisdom.weizmann.ac.il/~itsik/RC4/Papers/FluhrerMcgrew.pdf)*(PDF): 19–30. Archived from*[*the original*](http://www.wisdom.weizmann.ac.il/~itsik/RC4/Papers/FluhrerMcgrew.pdf)*(PDF) on 2 May 2014.*
  40. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-40) *Basu, Riddhipratim; Ganguly, Shirshendu; Maitra, Subhamoy; Paul, Goutam (2008). "A Complete Characterization of the Evolution of RC4 Pseudo Random Generation Algorithm". Journal of Mathematical Cryptology.****2****(3): 257–289.*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.1515/JMC.2008.012*](https://doi.org/10.1515%2FJMC.2008.012)*.*[*S2CID*](https://en.wikipedia.org/wiki/S2CID_(identifier))[*9613837*](https://api.semanticscholar.org/CorpusID:9613837)*.*
  41. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-41) *Fluhrer, Scott R.; Mantin, Itsik; Shamir, Adi (2001).*[*"Weaknesses in the Key Scheduling Algorithm of RC4"*](https://web.archive.org/web/20040602051734/http:/www.wisdom.weizmann.ac.il/~itsik/RC4/Papers/Rc4_ksa.ps)*.*[*Selected Areas in Cryptography*](https://en.wikipedia.org/wiki/Selected_Areas_in_Cryptography)*: 1–24. Archived from*[*the original*](http://www.wisdom.weizmann.ac.il/~itsik/RC4/Papers/Rc4_ksa.ps)*on 2 June 2004.*
  42. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-42) [*"Interim technology for wireless LAN security: WPA to replace WEP while industry develops new security standard"*](https://archive.today/20120709120158/http:/findarticles.com/p/articles/mi_m0DIS/is_2003_Jan/ai_n27590035/)*. Archived from*[*the original*](http://findarticles.com/p/articles/mi_m0DIS/is_2003_Jan/ai_n27590035/)*on 9 July 2012.*
  43. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-43) [*"RC4-drop(nbytes) in the Standard Cryptographic Algorithm Naming database"*](http://www.users.zetnet.co.uk/hopwood/crypto/scan/cs.html#RC4-drop)*.*
  44. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-44) *Rivest, Ron.*[*"RSA Security Response to Weaknesses in Key Scheduling Algorithm of RC4"*](https://www.rsa.com/rsalabs/node.asp?id=2009)*.*
  45. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-45) A. Klein, Attacks on the RC4 stream cipher, Designs, Codes and Cryptography (2008) 48:269–286
  46. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-46) Erik Tews, Ralf-Philipp Weinmann, Andrei Pyshkin. [Breaking 104-bit WEP in under a minute](https://eprint.iacr.org/2007/120).
  47. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-47) [Souradyuti Paul](https://en.wikipedia.org/wiki/Souradyuti_Paul" \o "Souradyuti Paul) and [Bart Preneel](https://en.wikipedia.org/wiki/Bart_Preneel), A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher. [Fast Software Encryption](https://en.wikipedia.org/wiki/Fast_Software_Encryption) – FSE 2004, pp245 – 259 [(PDF)](http://www.cosic.esat.kuleuven.be/publications/article-40.pdf).
  48. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-48) *John Leyden (15 March 2013).*[*"HTTPS cookie crypto CRUMBLES AGAIN in hands of stats boffins"*](https://www.theregister.co.uk/2013/03/15/tls_broken/)*. The Register.*
  49. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-49) *AlFardan; et al. (8 July 2013).*[*"On the Security of RC4 in TLS and WPA"*](http://www.isg.rhul.ac.uk/tls/RC4biases.pdf)*(PDF). Information Security Group, Royal Holloway, University of London.*
  50. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-50) [*"On the Security of RC4 in TLS and WPA"*](http://www.isg.rhul.ac.uk/tls/)*. Information Security Group, Royal Holloway, University of London. Retrieved 6 September 2013.* (website)
  51. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-51) [*"RC4 must die"*](http://www.isg.rhul.ac.uk/tls/RC4mustdie.html)*.*
  52. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-52) [*"Briefings - March 26 & 27"*](https://www.blackhat.com/asia-15/briefings.html#bar-mitzva-attack-breaking-ssl-with-13-year-old-rc4-weakness)*. 2015. Retrieved 19 November2016.*
  53. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-53) [*"Attacking SSL when using RC4"*](https://www.imperva.com/docs/HII_Attacking_SSL_when_using_RC4.pdf)*(PDF). 2015. Retrieved 19 November 2016.*
  54. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-rc4nomore_54-0) *Mathy Vanhoef and Frank Piessens (9 August 2015).*[*"RC4 NOMORE: Numerous Occurrence MOnitoring & Recovery Exploit"*](http://www.rc4nomore.com/)*.*
  55. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-55) *Ilya Mironov (1 June 2002),*[*"(Not So) Random Shuffles of RC4"*](http://eprint.iacr.org/2002/067)*,*[*Advances in Cryptology – CRYPTO 2002*](http://www.iacr.org/archive/crypto2002/24420305/24420305.pdf)*(PDF), Lecture Notes in Computer Science,****2442****, Springer-Verlag, pp. 304–319,*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.1007/3-540-45708-9\_20*](https://doi.org/10.1007%2F3-540-45708-9_20)*,*[*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier))[*978-3-540-44050-5*](https://en.wikipedia.org/wiki/Special:BookSources/978-3-540-44050-5)*, Cryptology ePrint Archive: Report 2002/067, retrieved 4 November2011*
  56. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-56) *[Souradyuti Paul](https://en.wikipedia.org/wiki/Souradyuti_Paul" \o "Souradyuti Paul);*[*Bart Preneel*](https://en.wikipedia.org/wiki/Bart_Preneel)*(2004),*[*"A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher"*](http://homes.esat.kuleuven.be/~psourady/publication-info/PP04-bias_rc4.htm)*, Fast Software Encryption, FSE 2004, Lecture Notes in Computer Science,****3017****, Springer-Verlag, pp. 245–259,*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.1007/978-3-540-25937-4\_16*](https://doi.org/10.1007%2F978-3-540-25937-4_16)*,*[*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier))[*978-3-540-22171-5*](https://en.wikipedia.org/wiki/Special:BookSources/978-3-540-22171-5)*, retrieved 4 November 2011*
  57. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-57) *Alexander Maximov (22 February 2007),*[*Two Linear Distinguishing Attacks on VMPC and RC4A and Weakness of RC4 Family of Stream Ciphers*](http://eprint.iacr.org/2007/070)*, Cryptology ePrint Archive: Report 2007/070, retrieved 4 November 2011*
  58. ^ [Jump up to:***a***](https://en.wikipedia.org/wiki/RC4#cite_ref-nec_58-0) [***b***](https://en.wikipedia.org/wiki/RC4#cite_ref-nec_58-1) *Yukiyasu Tsunoo; Teruo Saito; Hiroyasu Kubo; Maki Shigeri; Tomoyasu Suzaki; Takeshi Kawabata (2005),*[*The Most Efficient Distinguishing Attack on VMPC and RC4A*](http://www.ecrypt.eu.org/stream/papersdir/037.pdf)*(PDF)*
  59. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-59) *Bartosz Zoltak (2004),*[*"VMPC One-Way Function and Stream Cipher"*](http://www.vmpcfunction.com/vmpc.pdf)*(PDF),*[*Fast Software Encryption, FSE 2004*](https://www.iacr.org/archive/fse2004/30170209/30170209.pdf)*(PDF), Lecture Notes in Computer Science,****3017****, Springer-Verlag, pp. 210–225,*[*CiteSeerX*](https://en.wikipedia.org/wiki/CiteSeerX_(identifier))[*10.1.1.469.8297*](https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.469.8297)*,*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.1007/978-3-540-25937-4\_14*](https://doi.org/10.1007%2F978-3-540-25937-4_14)*,*[*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier))[*978-3-540-22171-5*](https://en.wikipedia.org/wiki/Special:BookSources/978-3-540-22171-5)*, retrieved 4 November2011*
  60. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-maximov_60-0) [*"CryptoLounge: RC4A"*](https://web.archive.org/web/20111001012601/http:/www.cryptolounge.org/wiki/RC4A)*. Archived from*[*the original*](http://www.cryptolounge.org/wiki/RC4A)*on 1 October 2011. Retrieved 4 November 2011.*
  61. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-rc4+_61-0) *Subhamoy Maitra; Goutam Paul (19 September 2008),*[*"Analysis of RC4 and Proposal of Additional Layers for Better Security Margin"*](http://eprint.iacr.org/2008/396)*,*[*Progress in Cryptology – INDOCRYPT 2008*](http://eprint.iacr.org/2008/396.pdf)*(PDF), Lecture Notes in Computer Science,****5365****, Springer-Verlag, pp. 27–39,*[*CiteSeerX*](https://en.wikipedia.org/wiki/CiteSeerX_(identifier))[*10.1.1.215.7178*](https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.215.7178)*,*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.1007/978-3-540-89754-5\_3*](https://doi.org/10.1007%2F978-3-540-89754-5_3)*,*[*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier))[*978-3-540-89753-8*](https://en.wikipedia.org/wiki/Special:BookSources/978-3-540-89753-8)*, Cryptology ePrint Archive: Report 2008/396, retrieved 4 November 2011*
  62. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-62) *Debjyoti Bhattacharjee; Anupam Chattopadhyay.*[*"Hardware Accelerator for Stream Cipher Spritz"*](https://blogs.ntu.edu.sg/debjyoti001/files/2016/07/paper-1rf0pxd.pdf)*(PDF). Secrypt 2016. Retrieved 29 July 2016.*
  63. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-63) *Banik, Subhadeep; Isobe, Takanori (20 March 2016). Peyrin, Thomas (ed.).*[*Cryptanalysis of the Full Spritz Stream Cipher*](https://semanticscholar.org/paper/18b763d71b21558232bca8335338400159d8e492)*. Lecture Notes in Computer Science. Springer Berlin Heidelberg. pp. 63–77.*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.1007/978-3-662-52993-5\_4*](https://doi.org/10.1007%2F978-3-662-52993-5_4)*.*[*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier))[*9783662529928*](https://en.wikipedia.org/wiki/Special:BookSources/9783662529928)*.*[*S2CID*](https://en.wikipedia.org/wiki/S2CID_(identifier))[*16296315*](https://api.semanticscholar.org/CorpusID:16296315)*.*
  64. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-64) Hongjun Wu, "The Misuse of RC4 in Microsoft Word and Excel". <https://eprint.iacr.org/2005/007>
  65. [**^**](https://en.wikipedia.org/wiki/RC4#cite_ref-65) [*"Skype's encryption procedure partly exposed"*](https://web.archive.org/web/20100711062958/http:/www.h-online.com/security/news/item/Skype-s-encryption-procedure-partly-exposed-1034577.html)*. www.h-online.com. Archived from*[*the original*](http://www.h-online.com/security/news/item/Skype-s-encryption-procedure-partly-exposed-1034577.html)*on 11 July 2010. Retrieved 8 July 2010.*

Further reading[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=25)]

* *Paul, Goutam; Subhamoy Maitra (2011).*[*RC4 Stream Cipher and Its Variants*](http://www.crcpress.com/product/isbn/9781439831359)*. CRC Press.*[*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier))[*9781439831359*](https://en.wikipedia.org/wiki/Special:BookSources/9781439831359)*.*
* [*Schneier, Bruce*](https://en.wikipedia.org/wiki/Bruce_Schneier)*(1995).*[*"Chapter 17 – Other Stream Ciphers and Real Random-Sequence Generators"*](https://www.schneier.com/book-applied.html)*. Applied Cryptography: Protocols, Algorithms, and Source Code in C (2nd ed.). Wiley.*[*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier))[*978-0471117094*](https://en.wikipedia.org/wiki/Special:BookSources/978-0471117094)*.*

External links[[edit](https://en.wikipedia.org/w/index.php?title=RC4&action=edit&section=26)]

* [Original posting of RC4 algorithm to Cypherpunks mailing list](http://cypherpunks.venona.com/archive/1994/09/msg00304.html), [Archived version](https://web.archive.org/web/20080207125928/http:/cypherpunks.venona.com/archive/1994/09/msg00304.html)
* [RFC 4345](https://tools.ietf.org/html/rfc4345) – Improved Arcfour Modes for the Secure Shell (SSH) Transport Layer Protocol
* [RFC 6229](https://tools.ietf.org/html/rfc6229) – Test Vectors for the Stream Cipher RC4
* [RFC 7465](https://tools.ietf.org/html/rfc7465) – Prohibiting RC4 Cipher Suites
* *Kaukonen; Thayer.*[*A Stream Cipher Encryption Algorithm "Arcfour"*](https://tools.ietf.org/html/draft-kaukonen-cipher-arcfour-03)*. I-D draft-kaukonen-cipher-arcfour-03.*
* [SCAN's entry for RC4](http://www.users.zetnet.co.uk/hopwood/crypto/scan/cs.html#RC4)
* [Attacks on RC4](https://web.archive.org/web/20150221230645/http:/www.wisdom.weizmann.ac.il/~itsik/RC4/rc4.html) at the [Wayback Machine](https://en.wikipedia.org/wiki/Wayback_Machine" \o "Wayback Machine) (archived 21 February 2015)
* [RC4 – Cryptology Pointers by Helger Lipmaa](http://kodu.ut.ee/~lipmaa/crypto/link/stream/rc4.php)
* [RSA Security Response to Weaknesses in Key Scheduling Algorithm of RC4](https://www.emc.com/emc-plus/rsa-labs/historical/rsa-security-response-weaknesses-algorithm-rc4.htm)

**RC4 in WEP**

* [(in)Security of the WEP algorithm](http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html)
* *Fluhrer; Mantin; Shamir (Summer–Fall 2002).*[*"Attacks On RC4 and WEP"*](https://web.archive.org/web/20150102135713/http:/www.wisdom.weizmann.ac.il/~itsik/RC4/Papers/rc4_wep.ps)*. CryptoBytes.****5****(2). Archived from*[*the original*](http://www.wisdom.weizmann.ac.il/~itsik/RC4/Papers/rc4_wep.ps)*(PostScript) on 2 January 2015.*

RC4 No Longer Secure — Check your HTTPS site now

[](https://medium.com/@andygambles?source=post_page-----cdb8fb0ae3a8----------------------)

[Andy Gambles](https://medium.com/@andygambles?source=post_page-----cdb8fb0ae3a8----------------------)

Follow

[Jul 16, 2015](https://medium.com/servertastic/rc4-no-longer-secure-check-your-https-site-now-cdb8fb0ae3a8?source=post_page-----cdb8fb0ae3a8----------------------) · 2 min read

At Servertastic we make the ability to purchase an [SSL Certificate](https://www.servertastic.com/ssl/) as simple as we possibly can. But as an installer you have many things you need to consider to ensure your site uses the best encryption available to secure the HTTPS site.

When connecting to a website using HTTPS there are several encryption algorithms that can be used. The client and server negotiate with each other to find one they both mutually support.

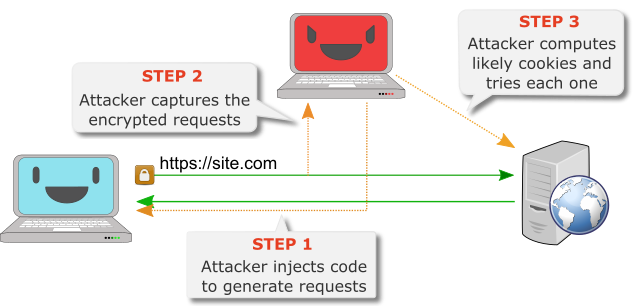
As computing power increases these algorithms come under regular attack and eventually become too weak to be considered secure. This is the case with RC4.

Google Chrome already highlights to when a site is using RC4 by stating [Your connection to [domain] is encrypted with obsolete cryptography](https://support.servertastic.com/your-connection-to-domain-is-encrypted-with-obsolete-cryptography/).

Recently security researches have demonstrated a plausible attack against a HTTPS website that utilised RC4. They were able to monitor traffic until they had enough data to break the encryption. This took a mere 52 hours.

[caption id=”attachment\_2843" align=”aligncenter” width=”640"]





via [RC4 NOMORE](http://www.rc4nomore.com/)[/caption]

The demonstration allowed the attackers to access decrypted cookie data for the website. However the attack was not limited to just cookies. Any system using RC4 encryption is vulnerable including WPA-TKIP Wifi networks.

**Is my website vulnerable?**

To check if your site has RC4 enabled we recommend using [SSL Labs Server Test](https://www.ssllabs.com/ssltest/). This will highlight any issues with your HTTPS implementation.

**How to disable RC4**

For Apache configurations you should add !RC4 to the SSLCipherSuite directive. For Windows users please refer to the following article: [Microsoft security advisory: Update for disabling RC4](https://support.microsoft.com/en-us/kb/2868725)

**Further Information**

* [RC4 NOMORE Website](https://www.rc4nomore.com/)
* [Breaking RC4 in WPA-TKIP and TLS Paper](http://www.rc4nomore.com/vanhoef-usenix2015.pdf)