

RC4 stream cipher and possible attacks on WEP

Lazar Stošić

College for professional studies educators
Aleksinac, Serbia

Milena Bogdanović

Teacher Training Faculty
University of Niš
Vranje, Serbia

Abstract—In this paper we analyze and present some weaknesses and possible attacks on the RC4 stream cipher which were published in many journals. We review some advantages and disadvantages which come from several authors, as well as similarities and differences which can be observed in the published results. Also, we analyze the Key Scheduling Algorithm (KSA) which derives the initial state from a variable size key, and strengths and weaknesses of the RCS stream cipher. Using examples from other papers, we show that RC4 is completely insecure in a common mode of operation which is used in the widely deployed Wired Equivalent Privacy protocol (WEP, which is part of the 802.11 standard).

Keywords—RC4 stream cipher; KSA; WEP; security of WEP; WEP attack.

I. INTRODUCTION

RC4, a fast output-feedback cipher, is one of the most widely used cryptosystems on the Internet, commonly used as the default cipher for SSL/TLS connections [20]. It was designed by Ron Rivest in 1987 for RSA Data Security, Inc., and kept as a trade secret until it leaked out in 1994 and is now available for public analysis [18]. RC4 is currently being standardized by the IETF under the name “Arcfour” [23]. RSA DSI did not confirm that the published algorithm is in the RC4 algorithm, but experimental tests showed that it produces the same outputs as the RC4 software. The RC4 key stream generation algorithm updates the RC4 internal state and generates one byte of key stream. The key stream is XORed to the plaintext to generate the ciphertext. RC4 is comprised of two algorithms: the Key Scheduling Algorithm (KSA) which turns a random key (whose typical size is 40-256 bits) into an initial permutation S of $\{0, \dots, N-1\}$, which uses the secret key to create a pseudo-random initial state, and the Pseudo Random Generation Algorithm (PRGA), which generates the pseudo-random stream to generate a pseudo-random output sequence. Both algorithms are presented in Fig. 1.

KSA(K) Initialization: For $i = 0$ to $N - 1$ $S[i] = i$ $j \leftarrow 0$ Scrambling: For $i = 0$ to $N - 1$ $j \leftarrow j + S[i] + K[i \bmod l]$ Swap($S[i], S[j]$)	PRGA(S) Initialization: $i \leftarrow 0$ $j \leftarrow 0$ Generation loop: $i \leftarrow i + 1$ $j \leftarrow j + S[i]$ Swap($S[i], S[j]$) Output $S[S[i] + S[j]]$
---	---

Figure 1. The RC4 Algorithms

(The Key Scheduling Algorithm and the Pseudo-Random Generation Algorithm)

In practical applications, stream ciphers are used with a session key which is derived from a shared secret key and an Initial Value (IV, which is transmitted unencrypted). The derivation of the session key can be done in various ways, such as concatenated after the IV.

Section I is the introduction to this paper. Section II presents the features of RC4 family ciphers, strengths and weaknesses of the RC4 stream cipher and existing attack methods aimed at them.

Section III shows the Wired Equivalent Privacy protocol, used for encrypting wirelessly transmitted packets on IEEE 802.11 networks.

Section IV presents discussion of what this study has shown, strengths and weaknesses of the methods, how the results support the current literature or refute current knowledge and their impact on current thinking or practice. Section V concludes this paper.

II. RC4 STREAM CIPHER

RC4 has a secret internal state which is a permutation of all $N=2n$ possible n bits words, along with two indices in it. In practical applications, $n=8$, and thus RC4, has a huge state of (1)

$$\log_2([256]^2 \cdot |S_{256}|) = \log_2(2^{16} \cdot 256!) \approx 1700 \text{ bits} \quad (1)$$

The initial state is derived from a variable-size key by a Key-Scheduling Algorithm (KSA), and then RC4 alternately modifies the state (by exchanging two out of the N values) and produces an output (by picking one of the N values).

RC4's internal state consists of a 256-byte array S , defining a permutation, as well as two integers $0 \leq i, j \leq 255$ acting as pointers into the array.

The RC4 key setup initializes the internal state using a key K of up to 256 bytes. RC4 keys are 2048 bits long, and their internal state consists of two counters i and j (each within $0 \leq i, j \leq 255$) plus an array of 256 8-bit bytes, called the S-box.

The S-box is initialized using the key K as follows (2):

```
for i = 0 to 255
    S(i) = i
    j = 0

for i = 0 to 255
    j = (j + S(i) + K(i)) mod 256
    swap S(i) and S(j)

i = 0
j = 0
```

(2)

Each next byte b of the keystream is produced using (3):

```
i = (i + 1) mod 256
j = (j + S(i)) mod 256
swap S(i) and S(j)
b = S(S(i) + S(j)) mod 256
```

(3)

For shorter key, the key is repeated as many times as necessary to fill the 2048-bit key. Once the S-box is initialized with the key, the RC4 algorithm is a loop that updates the internal state of the S-box and returns a byte of keystream. RC4 only protects the secrecy of a message, not its integrity. Other measures, such as the use of cryptographic checksums, are commonly used along with RC4.

RC4 stream cipher is used to protect internet traffic as part of the SSL (Secure Socket Layer) and TLS (Transport Layer Security) protocols, and to protect wireless networks as part of the WEP (Wired Equivalent Privacy) and WPA (Wi-Fi Protected Access) protocols. This attack was described by Fluhrer, Mantin and Shamir. It is a symmetric key algorithm and it is an important class of encryption algorithms. They encrypt individual characters (usually binary digits) of a plaintext message one at a time, using a simple time-dependent encryption transformation. The Blum-Goldwasser probabilistic public-key encryption scheme described in [7] is an example of asymmetric stream cipher. The same algorithm is used for both encryption and decryption as the data stream is simply XORed with the generated key sequence. The key stream is completely independent of the plaintext used. It uses a variable length key from 1 to 256 bits to initialize a 256-bit state table. The state table is used for Subsequent generation of pseudo-random bits and then to generate a pseudo-random stream which is XORed with the plaintext to give the cipher text.

The steps for RC4 encryption algorithm are as follows [17] and [16]:

- 1) Get the data to be encrypted and the selected key.
- 2) Create two string arrays.
- 3) Initiate one array with numbers from 0 to 255.
- 4) Fill the other array with the selected key.
- 5) Randomize the first array depending on the array of the key.
- 6) Randomize the first array within itself to generate the final key stream.

- 7) XOR the final key stream with the data to be encrypted to give cipher text.

One of the weaknesses of RC4 initialization mechanism is a major statistical bias in the distribution of the first output words. This bias makes it trivial to distinguish between several hundred short outputs of RC4 and random strings by analyzing their second word. This weakness can be used to mount a practical ciphertext-only attack on RC4 in some broadcast applications, in which the same plaintext is sent to multiple recipients under different keys. This unique statistical behavior is independent of the KSA, and remains applicable even when RC4 starts with a totally random permutation.

RC4 Strengths:

Some of RC4 Strengths [16]:

- 1) The difficulty of knowing which location in the table is used to select each value in the sequence.
- 2) A particular RC4 key can be used only once.
- 3) Encryption is about 10 times faster than DES.

RC4 Weaknesses:

Some of RC4 weaknesses [17] and [16]:

- 1) The RC4 algorithm is vulnerable to analytic attacks of the state table [6] and [15].
- 2) WEAK KEYS: these are keys identified by cryptanalysis that is able to find circumstances under which one or more generated bytes are strongly correlated with small subset of the key bytes. These keys can happen in one out of 256 keys generated [10], [5] and [15].

Many ways to break RC4 are classified as Distinguishing Attack, which makes use of the bias in output sequence. In 2004, some new stream ciphers were proposed, to which resistance to the attacks aimed at RC4 was added. They are exemplified by VMPC, a stream cipher proposed by B. Zoltak [4], and RC4A, an RC4 family algorithm improved by S. Paul and B. Preneel [11].

III. WIRED EQUIVALENT PRIVACY

Today, PC cards are most frequently used in home and business networks. All computers have a security protocol called Wired Equivalent Privacy (WEP). A device using an 802.11 card is configured with a key, that in practice usually consists of a password or a key derived from a password.

Wired Equivalent Privacy (WEP) is a protocol for encrypting wirelessly transmitted packets on IEEE 802.11 networks. In a WEP protected network, all packets are encrypted using the stream cipher RC4 under a common key, the root key R_k . R_k is the WEP or root key and IV is the initialization vector for a packet. $K = R_k \parallel IV$ is the session or per packet key. X is a key stream generated using K . The WEP protocol is designed to provide privacy to packet based wireless networks based on the 802.11b standard [19]. The WEP encrypts by taking a secret key and a per-packet 3 byte IV , and using the IV followed by the secret key as the RC4 key. The attacker is able to retrieve the first byte of the RC4 output from each packet.

WEP uses a 40-bit secret key (which was the largest easily exportable key when WEP was designed), shared between all the users and the network access point. For every packet, the sender chooses a new 24 bit Initialization Vector (IV), and the 64-bit RC4 key is the concatenation of the chosen IV (occurring first) and the shared key (occurring last). Such an IV-based mode of operation is commonly used in stream ciphers in order to generate different PRGA outputs from the same long term key, and the frequent resetting of the PRGA is designed to overcome the unreliable nature of the Wireless LAN environment.

The simplest weakness is the small size of the secret key and the IV: A 40-bit key can be recovered by an exhaustive search in less than one day. The limited size (224) of the IV space implies that IVs are reused during the encryption of different packets. This mode can be attacked by constructing a dictionary of all the 224 IVs along with their corresponding key streams. WEP defines no easy mechanism for changing the shared key, and thus the key is usually changed only infrequently, increasing the attacker's chance to construct this dictionary.

The first "real" attack makes it possible to derive an arbitrarily long key in time which grows only linearly with its length in the weakest attack model of known plaintext and IV developed in [12], and is outlined in the next section.

A first analysis of the design failures of the WEP protocol was published by Borisov, Goldberg and Wagner [9] in 2001, which showed that the IV merely protects against random errors but not against malicious attackers. They observed that old IV values could be reused, thus allowing to inject messages.

RC4 are specified. Several PC cards reset IVs to zero every time they are initialized, and then increment them by one for every use. This results in high likelihood that keystreams will be reused, leading to simple cryptanalytic attacks against the cipher, and decryption of message traffic.

Fluhrer, Mantin and Shamir presented a related key ciphertext-only attack against RC4 [13] as used in WEP. In WEP, the key scheduling algorithm uses either a 64-bit packet key (40-bit secret key plus 24-bit IV) or a 128-bit key (104-bit secret key plus 24-bit IV) to set up the RC4 state array, S , which is a permutation of $\{0, \dots, 255\}$. The output generator uses the state array S as well as two counters, i and j , to create a pseudorandom sequence.

In order for this attack to work, the IVs need to fulfill a so-called "resolved condition". This attack was suspected to be applicable to WEP, which was later demonstrated by Stubblefield et al. [1]. Approximately 4 million different frames need to be captured to mount this attack. Vendors reacted to this attack by filtering IVs fulfilling the resolved condition, so-called "weak IVs". On the other side, Klein [3] showed an improved way of attacking RC4 using related keys that does not need the "resolved condition" on the IVs and gets by with a significantly reduced number of frames.

A. The WEP Attack

The attack starts with the known IV as a basis, and repeatedly applies the sub-attack in order to recover all the keywords in the secret key SK. To conduct an attack, the cryptanalyst needs the first output word of a large number of RC4 streams along with the IV that was used to generate each one of them. Since in WEP the IVs are transmitted in the clear and the first message word in most packets is a known constant, these requirements are automatically satisfied.

With about 60 such IVs, the attacker can re-derive the key byte with reasonable probability of success. The number of packets required to obtain that number of IVs depends on the exact IVs that the sender uses. Although the 802.11b standard does not specify how an implementation should generate these IVs, common practice is to use a counter to generate them. We now analyze the performance of this attack for two different counter modes. If the counter does not start from zero, the attacker has an alternative strategy available to him. If the attacker assumes the first two bytes of secret key, then for each initial IV byte, there are approximately 4 settings of the remaining two bytes that set up the permutation as required to re-derive a particular key byte.

Fluhrer S., Mantin I. and Shamir A. in their work Attacks on RC4 and WEP explain that the first x words of the KSA key are known. This makes it possible to simulate the first x rounds of the KSA and compute the permutation S_{x-1} and the indices i_{x-1} and j_{x-1} at that point. The next value of i is also known ($i_x=x$) but the next value of j (j_x) depends on the unknown target keyword $K[x]$ (since $j_x=j_{x-1}+S_{x-1}[x]+K[x]$) and thus each of the values j_x and $K[x]$ can be easily derived from the other. Consequently, given $S_x[x]$, we can compute which value was in position j_x in the known permutation S_{x-1} , and by inverting this permutation, we can recover j_x itself.

IV. DISCUSSION

RC4 is a symmetric key algorithm. Stream cipher is an important class of encryption algorithms. They encrypt individual characters of a plaintext message one at a time, using a simple time-dependent encryption transformation. RC4 is comprised of two algorithms: the Key Scheduling Algorithm (KSA) which turns a random key (whose typical size is 40-256 bits) into an initial permutation S of $\{0, \dots, N-1\}$, which uses the secret key to create a pseudo-random initial state, and the Pseudo Random Generation Algorithm (PRGA), which generates the pseudo-random stream to generate a pseudo-random output sequence.

We see that some of RC4 strengths were: the difficulty of knowing which location in the table is used to select each value in the sequence; a particular RC4 key can be used only once; encryption is about 10 times faster than DES. On the other side, RC4 weaknesses were: The RC4 algorithm is vulnerable to analytic attacks of the state table; WEAK KEYS: these are keys identified by cryptanalysis that is able to find circumstances under which one or more generated bytes are strongly correlated with small subset of the key bytes.

In current literature we can see that many ways to break RC4 are classified as Distinguishing Attack. These ways make use of the bias in output sequence. The first “real” attack makes it possible to derive an arbitrarily long key in time which grows only linearly with its length in the weakest attack model of known plaintext and IV. A first analysis of the design failures of the WEP protocol showed that the IV merely protects against random errors but not against malicious attackers. They observed that old IV values could be reused, thus allowing to inject messages.

In the last few decades many stream ciphers have been proposed. Most of them are easy to implement on hardware but their performance is slow when implemented on software. Since RC4 is such a widely used stream cipher, it attracted considerable attention in the research community since it was proposed. The strength of the RC4 key does not grow linearly with the increase in the key length.

V. CONCLUSIONS

The main contribution of this paper is the presentation of the established and proven deficiencies of RC4 which are caused by its extreme simplicity. Based on the results of numerous research studies, we can conclude that the initialization of the pseudo-random index j to 0 seems to be the most problematic operation, and both the second byte bias and the IV weakness could be avoided by using a more complex initialization of j . Possible methods for initializing j are to use j from the end of the KSA or to give it the value of one of the key words. The invariance weakness and the IV weakness are inherent consequences of the structure of the KSA.

For the RC4 stream cipher, every key has a family of related keys which result in a substantially similar keystream. The strength of the RC4 key does not grow linearly with the increase in the key length. If RC4 is deployed using keys longer than the customary 128 bits, we advise discarding the first 256 bytes of the keystream.

A perfect initialization mechanism is not easy to achieve. A common mode of operation to achieve these contradicting goals is to discard a prefix of output bits. These mute rounds usually disconnect the generated stream from the initialization process, and improve the “randomness” of the generated stream.

The discarded prefix should also grow in the same way (exponentially) when enlarging RC4 words into 16 bits (which is sometimes recommended for faster encryption of large amount of data). The expression of the invariance weakness spreads over several hundred words in RC416 and eliminating only 256 words is not sufficient when N is larger. The reduced version RC46 can be attacked with practical complexity, while for stronger version (RC4 $n > 6$) it is possible to mount enhanced (but impractical) attacks.

REFERENCES

[1] A. Stubble, J. Ioannidis, and A. D. Rubin, “A key recovery attack on the 802.11b wired equivalent privacy protocol (WEP)”, ACM Transactions on Information and System Security, Volume 7 Issue 2, May 2004, pp. 319-332.

[2] A. Bittau, M. Handley, and J. Lackey, “The final nail in WEP’s coffin. In IEEE Symposium on Security and Privacy”, pp. 386-400, IEEE Computer Society, 2006.

[3] A. Klein, “Attacks on the RC4 stream cipher”, volume 48 Issue 3, pp. 269 – 286, Designs, Codes and Cryptography, September 2008. doi>10.1007/s10623-008-9206-6

[4] B. Zoltak: “VMPC One-Way Function and Stream Cipher,” Fast Software Encryption, FSE 2004, LNCS 3017, pp.210-225, Springer-Verlag, 2004.

[5] E. Biham and Y. Carmeli, “Efficient Reconstruction of RC4 Keys from Internal States”, FSE 2008, pp. 270-288, vol. 5086, Lecture Notes in Computer Science, Springer.

[6] G. Paul, S. Rathi and S. Maitra, “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, pp. 285-294 and Designs, Codes and Cryptography Journal, pp. 123-134, vol. 49, no. 1-3, December 2008.

[7] M. Biryukov, A. Shamir, and D. Wagner, “Real time cryptanalysis of A5/1 on a PC”, FSE: Fast Software Encryption, 2000., pp. 1-18.

[8] M. Akgun, P. Kavak, H. Demirci, “New Results on the Key Scheduling Algorithm of RC4”, INDOCRYPT 2008, pp. 40-52, vol. 5365, Lecture Notes in Computer Science, Springer.

[9] N. Borisov, I. Goldberg, and D. Wagner, “Intercepting mobile communications: the insecurity of 802.11”, In ACM MobiCom 2001, pp. 180-189. ACM Press, 2001.

[10] R. Basu, S. Maitra, G. Paul and T. Talukdar, “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), June 8-12, 2009, Tarragona, Spain, pp. 137-148, vol. 5527, Lecture Notes in Computer Science, Springer.

[11] S. Paul, and B. Preneel, “A New Weakness in the RC4 Keystream Generator,” Fast Software Encryption, FSE 2004, LNCS 3017, pp.245-259, Springer-Verlag, 2004.

[12] S. R. Fluhrer, I. Mantin, and A. Shamir, “Weaknesses in the key scheduling algorithm of RC4”, In SAC’2001, 2001.

[13] S. R. Fluhrer, I. Mantin, and A. Shamir, “Weaknesses in the key scheduling algorithm of RC4”, In Serge Vaudenay and Amr M. Youssef, editors, Selected Areas in Cryptography 2001, volume 2259 of Lecture Notes in Computer Science, pp. 1-24. Springer, 2001.

[14] S. Dorhofer, “Empirische Untersuchungen zur WLAN-Sicherheit mittels Wardriving”, Diplomarbeit, RWTH Aachen, September 2006. (in German).

[15] V. Tomašević, S. Bojanić, O. Nieto-Taladriz, “Finding an internal state of RC4 stream cipher”, Information Sciences, Volume 177, issue 7, 01. April, 2007, pp.1715-1727.

[16] W. Mao, Modern Cryptography Theory and Practice, Prentice Hall, New Jersey, 2004.

[17] W. Stilings, Cryptography and Network Security Principles and practices, Fourth Edition, PEARSON, USA, 2006.

[18] B. Schneier, Applied Cryptography, John Wiley and Sons, New York, 2nd edition, 1996.

[19] LAN/MAN Standard Committee, Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, 1999 edition, IEEE standard 802.11, IEEE Computer Society, 1999.

[20] T. Dierks and C. Allen, The TLS Protocol, Version 1.0, Internet Engineering Task Force, January 1999.

[21] E. Tews, R. P. Weinmann and A. Pyshkin, “Breaking 104 bit WEP in less than 60 seconds”, IACR Eprint Server, <http://eprint.iacr.org/2007/120.pdf>, number 2007/120, Accessed April 1, 2007.

[22] S. Fluhrer, I. Mantin, A. Shamir, “Attacks on RC4 and WEP, RSA Laboratories”, http://www.rsa.com/rsalabs/cryptobytes/cryptobytes_v5n2.pdf Volume 5, No. 2, Summer/Fall 2002

- [23] K. Kaukonen and R. Thayer, "A Stream Cipher Encryption Algorithm Arcfour", <http://tools.ietf.org/html/draft-kaukonen-cipher-arcfour-03> , Internet Engineering Task Force (IETF), July 1999.

AUTHORS PROFILE



Lazar Stošić received the Ph.D. degrees of computer sciences on Faculty of Informatics and Information Technology, Novi Pazar, Serbia. Professor on College for professional studies educators, Aleksinac, Serbia. Member of the Society for media and science, e-learning center at the University of Zurich since 2009. - Gesellschaft für Medien in der Wissenschaft (GMW) e. V., E-Learning Center der Universität Zürich, Zürich. Member of the Society for Computer Science, Germany since 01.07.2009. - Gesellschaft für Informatik e.V. (GI), German Informatics Society, Mitgliedsnummer: GI 59631 AhrstraBe 45, 53175 Bonn, Germany. His research interests include Information Technology, Computer System, Computer Education and Computer Distance Learning.



Milena Bogdanović is the assistant professor at the Teacher Training Faculty in Vranje, Serbia (major in Mathematics and Informatics – Mathematics 1, Mathematics 2, Elementary mathematical concepts, IT in Education, Educational technology, Elements of mathematics). She is the Reviewer of international journals - IJACSA – International Journal of Advanced Computer Science and Applications; Member of the editorial boards of international journals – International Journal of Computer Systems and Applications – International Scientific Press; Reviewer of the Book of solved tasks in Mathematics 2; Experience in teaching in secondary school and university; Participant in numerous seminars and training for educational reform, active learning, Lifelong Learning, Mathematics and Applications. She is the author of two books and of about 40 of published scientific papers in the field of the mathematics and computer science. Her professional papers discuss problems in the field of applications of multimedia in teaching, combinatorial optimization, genetic algorithms, directable automata.