

Weaknesses in the Key Scheduling Algorithm of **RC4**

Denisa Sandu

11th January, 2018

Introduction

În cadrul paperului analizat, al cărui autori sunt Scott Fluhrer, Itsik Mantin și Adi Shamir, sunt prezentate vulnerabilitățile din algoritmul de planificare a cheilor la RC4 și este descrisă semnificația acestor vulnerabilități în criptanaliză. Cea mai relevantă problemă pe acest subiect este lipsa securității în Wired Equivalent Privacy protocol (WEP), care reprezintă inițial un algoritm de criptare menit să ofere rețelelor de tip wireless LAN același nivel de securitate existent în wired LANs.

RC4 reprezintă cel mai folosit stream cipher în aplicațiile software și funcționează pe baza unei stări interne secrete care este o permutare a tuturor celor 2^n cuvinte de câte n biți, împreună cu 2 indici în ea. În aplicațiile practice se utilizează $n = 8$.

Key Scheduling Algorithm (KSA) obține starea inițială din chei de lungimi variabile și prezintă 2 vulnerabilități majore:

1. Existența unor clase mari de chei slabe, în care o mică parte din cheia secretă determină un număr mare de biți din permutarea inițială
2. Având în vedere că aceeași parte secretă a cheii este folosită împreună cu numeroase valori deja descoperite, un atacator este capabil să obțină partea secretă analizând outputul inițial

Descrierea RC4

Procesul este alcătuit din 2 părți:

1. Un algoritm de planificare a cheilor (KSA) care transformă o cheie random (de dimensiune 40-256 de biți) într-o permutare inițială a lui S
2. O parte de generare de output în care un PRGA folosește permutarea anterioară pentru a obține o secvență pseudo-random de output

<p>KSA(K)</p> <p>Initialization:</p> <p>For $i = 0 \dots N - 1$</p> <p style="padding-left: 40px;">$S[i] = i$</p> <p style="padding-left: 20px;">$j = 0$</p> <p>Scrambling:</p> <p>For $i = 0 \dots N - 1$</p> <p style="padding-left: 40px;">$j = j + S[i] + K[i \bmod \ell]$</p> <p style="padding-left: 40px;">$Swap(S[i], S[j])$</p>	<p>PRGA(K)</p> <p>Initialization:</p> <p style="padding-left: 40px;">$i = 0$</p> <p style="padding-left: 40px;">$j = 0$</p> <p>Generation loop:</p> <p style="padding-left: 40px;">$i = i + 1$</p> <p style="padding-left: 40px;">$j = j + S[i]$</p> <p style="padding-left: 40px;">$Swap(S[i], S[j])$</p> <p style="padding-left: 40px;">Output $z = S[S[i] + S[j]]$</p>
--	---

Aplicarea atacului in WEP

Protocolul WEP a fost creat pentru a oferi privacy in retelele wireless bazate pe packete in standardul 802.11. Criptarea se realizeaza luand un IV de 3 bytes si o cheie secreta pentru fiecare packet, pe care le utilizeaza drept cheie pentru RC4. Ulterior, transmite IV-ul si payloadul criptat primit de la RC4. Prin analizarea a suficient de multe packete criptate, se poate reconstrui cheia secreta.

In cazul in care atacatorul este capabil sa intercepteze primul octet din rezultatul RC4 pentru fiecare pachet, pentru a reconstitui byte-ul B din cheie, el are nevoie sa stie octetii anteriori si sa caute IV-uri din care se obtine o permutare, astfel incat:


$$X = S_{B+3}[1] < B + 3$$

$$X + S_{B+3}[X] = B + 3$$

Utilizand 60 astfel de IV-uri, atacatorul poate obtine octetul din cheie cu o probabilitate foarte mare de a avea succes. Numarul de pachete necesare pentru a obtine acel numar de IV-uri depinde strict de vectorii de initializare folositi de transmitator. In practica, se utilizeaza un contor pentru a ii genera.

Implementare

Pentru a implementa atacul, am utilizat un tool numit [arcfour](#) care genereaza fake traffic criptat utilizand WEP. Se genereaza aproximativ 48M de trafic, pe care se poate aplica algoritmul. Modul in care acesta functioneaza este urmatorul:



Atacul functioneaza doar asupra packetelor care au IV-ul de forma $(a + 2, N - 1, x)$, unde a este indicele byte-ului din cheie pe care incerca sa il ghiceasca, N este 256 iar x este o valoare irelevanta. WEP-ul utilizeaza IV-uri de 24 de biti.

S-ul, echivalent al permutarii generate de KSA, este initializat cu valori consecutive din intervalul $[0, N - 1]$ si asupra lui se aplica primele $a + 3$ iteratii din KSA. Atacatorul poate sa obtina astfel o valoare posibila pentru $K[i] = (O - j - S[i]) \bmod N$. Pentru a mari nivelul de certitudine, acesta trebuie sa repete algoritmul pentru cat mai multe IV-uri ce respecta conditia mentionata anterior si sa numere aparitiile octetilor din cheie obtinuti la fiecare pas. Octetul real din cheie ar trebui sa apara de un numar considerabil de ori mai des decat orice alt octet, motiv pentru care atacatorul poate sa ghiceasca cheia.

Ulterior, va seta byte-ul din cheie ca fiind ghicit si va repeta procedeul pentru urmatoorii 4 octeti.

Referinte

[1] [*Weaknesses in the Key Scheduling Algorithm of RC4*](#)

[2] [*Wireless Traffic Capture and Analysis*](#)

[3] [*Arcfour*](#)

[4] [*Fluhrer, Mantin and Shamir attack*](#)