Универзитет у Београду

Математички факултет

Катедра за рачунарство и информатику

Мастер рад

Напад на шифру RC4 у оквиру протокола WEP

Студент: Александра Арсић, 1025/2013

Ментор: др. Миодраг Живковић

Београд, септембар 2014.

Садржај

1. Увод

Од последњих деценија 20. века бежичне мреже се све више користе и IP (*Internet Protocol*) мрежа је приступачнија корисницима. Бежичне мреже користе радио сигнале за пренос података, те су подложне и прислушкивању за разлику од жичаних мрежа. Како је овај тип мрежне комуникације доступан и јавности, захтева се да се подаци који се на њој користе и шаљу буду заштићени. Комитет IEEE (*Institute of Electrical and Electronics Engineers*) је прописао скуп стандарда, IEEE 802.11 који регулише безбедност на WLAN-у (*Wireless Local Area Network*). У овом стандарду је описан и протокол WEP (*Wired Equivalent Privacy*) који за безбедност саобраћаја на мрежи користи алгоритам RC4 [4]. Алгоритам RC4 има неколико слабости које се преносе и на протокол WEP. Циљ овог рада је да се детаљније опишу и изложе поменути недостаци, као и то како се они могу искористити не би ли се открио кључ који се користи при шифровању порука пре слања преко ове мреже [3].

Рад је подељен на шест поглавља. Прво поглавље је уводно.

Друго поглавље описује и дефинише појмове који ће у раду бити коришћени. Описан је поступак шифроване комуникације, појам кључа и начин његове употребе. Описане су проточне шифре и начин на који функционишу.

Треће поглавље се састоји из три одељка. Први одељак представља алгоритам RC4, фазе из којих се алгоритам састоји и њихове карактеристике. Наредни одељак је посвећен протоколу WEP. Изложена је спецификација протокола WEP, описан поступак комуникације и структура оквира или пакета, који се користе у комуникацији. Трећи одељак повезује претходна два. У њему је описана употреба алгоритма RC4 у протоколу WEP, изложене су неке од мана овог алгоритма и начин на који су оне превазиђене у самом протоколу.

Четврто поглавље описује напад на RC4 у оквиру протокола WEP. У овом поглављу су објашњене математичка и статистичка позадина напада. У наредном одељку овог поглавља је на конкретном примеру дата илустрација напада.

У петом поглављу је изложена програмска реализација напада. Наведени су псеудокодови који детаљније описују идеју напада. Поглавље се састоји од три одељка. У првом одељку је описана структура и начин креирања пакета који су за напад потребни. У наредном је описан напад над прикупљеним подацима. Последњи одељак даје статистичке

анализе и обрађује резултате програма за напад који су се аутору учинили занимљивим и корисним.

У последњем, шестом поглављу дата су завршна разматрања и предлози за будућа истраживања и модификације имплементираног напада.

2. Уводни појмови

У овом поглављу је описан поступак шифроване комуникације и дата објашњења појмова која ће бити коришћена у раду.

Особа **А** жели да пошаље шифровану поруку особи **Б**, тако да буде сигурна да неко други, особа **С**, уколико је пресретне не може да зна њену садржину. Порука коју особа **А** треба да пошаље је **отворени текст**, на пример „Мастер рад“. **Шифрат** је шифрована порука која се шаље, на пример „Абракадабра“.

Операција коју особа **А** примењује, да би од отвореног текста добила шифровани, назива се **шифровање**. Шифровање представља трансформацију отвореног текста помоћу функције чији су параметри задати кључем. Особа **В** по пријему поруке, треба да изврши операцију **дешифровања** како би прочитала њен садржај. Текст који треба шифровати може се посматрати као низ бинарно записаних **ASCII** кодова. **Кодирање** трансформише низ карактера отвореног текста у низ битова. **Декодирање** трансформише низ цифара или битова у карактере отвореног текста.

Алгоритам шифровања и алгоритам дешифровања зависе од посебног параметра који се зове **тајни кључ**. Тајни кључ је познат особи **А** која поруку шаље, као и особи **Б** која је примила поруку. Уколико обе особе користе исти тајни кључ, реч је о **симетричном систему.** Кључ се тада назива симетрични заједнички кључ.

Криптоанализа је процес који особа **С** покушава да изврши над подацима које је пресрела. Не знајући тајни кључ, особа **С** покушава да сазна отворени текст. Успешна криптоанализа се назива **декриптирање**. О претходно наведеним појмовима се више може прочитати у литератури [1].

Бежична приступна тачка (*Wireless access point – WAP*) је бежични уређај који спаја бежичне уређаје са жичаним мрежама користећи стандарде попут WI-FI (*Wireless-Fidelity*).

Углавном се повезује са рутером, управља доделом радио канала и посредује у преносу података између повезаних корисника [11]. **Протокол** представља скуп правила и конвенција за слање информација преко мреже[13].

2.1. Појам кључа и начин изградње

Често се дешава да је дужина текста који се шифрује доста већа од дужине тајног кључа којим се шифрује. У том случају се генерише **низ кључа** (*keystream*) периодичним понављањем тајног кључа (*key*). За исти одабир тајног кључа се увек добија исти низ кључа.

Уколико нападач у једном моменту дође у посед и отвореног и шифрованог текста, с лакоћом може сазнати и тајни кључ којим се шифровање обавља. Даље ће лако откривати отворени текст при сваком пресретању шифрованог. Другим речима, обављаће исти поступак као и прималац поруке. Због тога је добра пракса с времена на време, обновити кључ.

Алтернатива је употреба **иницијаног вектора** (IV) који би се мењао при сваком шифровању. Тајни кључ би у овом случају остао непромењен и настављао би се на IV. Проширени тајни кључ називамо **кључ**. Низ кључа настаје периодичним понављањем кључа, а не као раније тајног кључа. Овим је избегнута честа промена тајног кључа, али и обезбеђена већа сигурност.

2.2. Проточне шифре

Проточне шифре трансформишу отворени текст симбол по симбол, тј. бит по бит. Нека су m_1, m_2, m_3 … битови поруке коју треба шифровати. Битове низа кључа којим се шифровање врши означимо са k_1, k_2, k_3 … и они су познати и особи **A**, која шифрује поруку, и особи **B** која ће поруку дешифровати. Резултат процеса шифровања је шифровани текст, чије битове ћемо означавати са c_1, c_2, c_3 …

Поступак шифровања може се описати изразом:

$$m_i \oplus k_i = c_i.$$

Слично, поступак дешифровања се може описати изразом:

$$c_i \oplus k_i = m_i .$$

Наведен пример шифровања и дешифровања се сврстава у самохронишуће шифре. Наиме, уколико су нам познати кључ и један од отвореног и шифрованог текста онда применом наведених формула на одговарајуће парове битова, можемо сазнати и трећи, непознати низ. Операција **XOR** ($\oplus$) је захвална за коришћење јер се идентично примењује и у поступку шифровања и дешифровања.

Главна особина проточних шифара је та што се једноставном применом XOR операције лако долази до једног од три елемената на основу преостала два.

3. Алгоритам RC4 и протокол WEP

У овом поглављу је описан алгоритам RC4, као и фазе из којих се алгоритам састоји. Предочене су неке од слабости алгоритма, а у наредном одељку је изложен протокол WEP као и начин комуникације. Такође, описани су поступци шифровања и дешифровања. Последње поглавље наводи специфичности алгоритма RC4 у оквиру протокола WEP.

3.1. Алгоритам RC4

RC4 је најчешће коришћена проточна шифра, креирана 1987. године [8]. Њен творац је Р. Рајвест (*R. Rivest* [2]). До 1994. године њен алгоритам није био познат јавности. Њена употребна вредност се није смањила током протеклих година, она се и данас користи за шифровање саобраћаја на интернету у мрежним протоколима као што су SSL (*Secure Sockets Layer*), TLS (*Transport Layer Security*), WEP (*Wired Equivalent Privacy*), WPA (*Wi-Fi Protected Access*) и другим. Алгоритам се примењује и на Microsoft Windows, Lotus Notes, Apple Open Collaboration Environment (*AOCE*) и Oracle Secure SQL.

Циљ алгоритма RC4 је генерисање псеудослучајног низа бајтова којим ће се шифровати отворени текст пре слања. Постоје разне варијанте алгоритма које имају различиту вредност параметра n. Највише се користи варијанта са вредношћу $n = 8$. Основна компонента алгоритма RC4 је променљива пермутација S скупа $\{0,1,\dots,N-1\}$, где је $N = 2^n$. Све операције алгоритма RC4 које су повезане са индексима пермутације S се рачунају по модулу N.

Алгоритам RC4 се састоји из две фазе. Прва **фаза KSA** (*Key-scheduling algorithm*) иницијализује пермутацију S у зависности од кључа. Следећа је **фаза PRGA** (*Pseudo-random generation algorithm*). Циљ фазе PRGA јесте да промени пермутације S и из ње одабере низ бајтова намењен XOR-овању са бајтовима отвореног текста.

3.1.1. Фаза KSA

Улазни параметар за фазу KSA је проширени кључ K дужине N бајтова. На почетку ове фазе иницијализују се елементи **пермутације *S*** тако да сваки елемент добије вредност позиције, тј. индекса на коме се налази. Следећи корак је „мешање“ елемената низа S. Овај корак се одвија у тачно N итерација чиме се постиже приступање сваком елементу низа бар једном. У итерацијама $i = 0, 1, \dots, N-1$ одређује се индекс j елемента са којим се замењује елемент низа S са индексом i. Индекс j не зависи само од тренутног стања низа S, већ зависи и од кључа K. За различите кључеве добијају се различите пермутације S, али за исти кључ добија се увек иста пермутација.

Прецизније се фаза KSA описује следећим псеудокодом.

Ulazni parametri: $Tajni\ ključ\ K[0 \dots N-1]$
Izlazne vrednonosti: $Permutacija\ S$

$//\ Inicijalizacija$:
for $i \leftarrow 0$ ***to*** $N-1$ ***do***
    $S[i] = i;$
    $j = 0;$
$//\ Mešanje\ elemenata\ permutacije$:
for $i \leftarrow 0$ ***to*** $N-1$ ***do***
    $j = j + S[i] + K[i];$
    $Zameni\ mesta\ (S[i], S[j]);$
$//\ Priprema\ za\ funkciju\ PRGA$:
    $j = 0;$
    $i = 0;$

3.1.2. Фаза PRGA

Пошто је пермутација S одређена, приступа се другој фази, фази PRGA. Излаз функције PRGA је код који генерише задати број бајтова, тј. карактери са којим ће бити XOR-овани бајтови отвореног текста. Пре првог корака фазе PRGA, иницијализују се променљиве i и j на 0. Током шифровања текста, функција се позива докле год има бајтова отвореног текста који нису шифровани. Променљива i се увећава на предвидљив начин, док променљива j зависи од тренутне вредности пермутације S. Након одређивања вредности променљивих i и j врши се замена елемената у низу S на тим позицијама и рачуна се индекс t елемента који је излазна вредност текућег корака.

Псеудокод једног корака функције PRGA је изложен у наставку:

Ulazni parametri: $Permutacija\ S[0 \dots N-1]\ koja\ zavisi\ od\ tajnog\ ključa,$ indeksi i i j
Izlazne vrednonosti: $Pseudo\ slučajni\ element\ permutacije\ S$

$i = i + 1;$

$j = j\ + S[i];$

$Zameni\ mesta\ (S[i], S[j]);$

$t = S[i] + S[j]\ ;$

return $z = S[t];$

Један од недостатака алгоритма RC4 је тај што нападач може да предвиди излазни бајт из прве итерације функције PRGA. Бројач i увек има вредност 1, док се j поставља на S[1], детаљније се о овој тврдњи може погледати у литератури [3]. Уколико је нападачу, поред првог бајта шифрованог текста, познат и први бајт отвореног текста он може израчунати њиховим XOR-овањем бајт који је добијен приликом првог позива функције

PRGA. Нека је то бајт X. Нападач сада зна да се у пермутацији S бајт са вредношћу X налазио на позицији $t = S[1] + S[S[1]]$.

3.2. Спецификација протокола WEP

Стандард *IEEE 802.11* WEP обезбеђује аутентикацију и шифровање података између рачунара и бежичне приступне тачке користећи приступ симетричног заједничког кључа [10]. Овај протокол треба да бежичним локалим мрежама понуди безбедност какву имају жичане локалне мреже [5]. Бежичне мреже се физички не могу заштити јер је простирање сигнала тешко ограничити. Протокол WEP се користио на старијим уређајима како би се заштитила комуникација између клијента и приступног уређаја, али је због криптографске слабости, која је описана у овом раду, замењен протоколом WPA (*Wi-Fi Protected Access*).

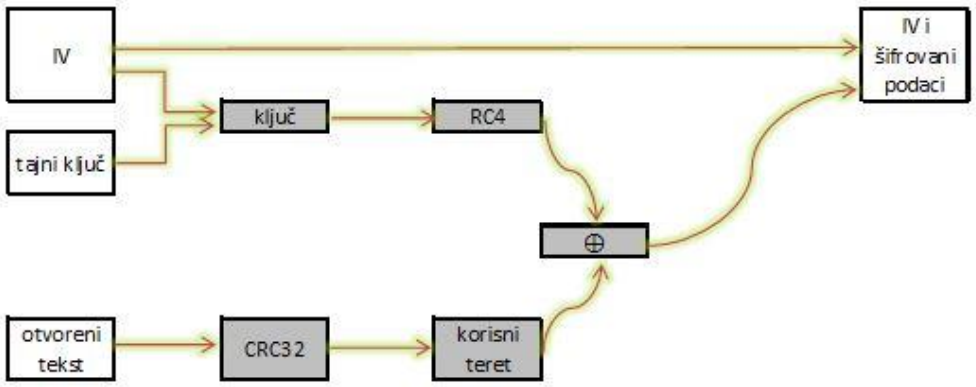
Прва верзија протокола WEP, објављена 1997. године, је користила 40-битни тајни кључ који је познат свим корисницима мреже. За сваки пакет, пошиљалац бира нови 24-битни иницијални вектор и на њега надовезује 40 бита тајног кључа. Касније је протокол измењен у смислу дужине тајног кључа који се надовезује на иницијални вектор. Дужина кључа је са 40 повећана на 104 бита. Било да се ради о 64-битном или 128-битном кључу у оквиру протокола WEP, алгоритами шифровања и дешифровања се не разликују.

Приступну тачку бежичних комуникација чине примопредајни уређаји и одговарајућа комуникациона опрема која има функцију да је повеже са осталим деловима комуникационе мреже. Начин размене кључева није дефинисан стандардом. Свака приступна тачка на мрежи добија тајни кључ који дели са базном станицом. Кључеви могу на пример, бити уграђени у производ од стране произвођача или размењени жичаном мрежом. Постоји могућност да свака приступна тачка или кориснички рачунар насумично одабере кључ, потом шифрује јавним кључем друге стране и пошаље бежичним путем.

Протокол WEP за шифровање користи алгоритам RC4. Подаци који се транспортују су складиштени у структуре који се називају **пакети** или **оквири**. Пакети који се шаљу кроз мрежу се састоје од заглавља и корисног терета. **Заглавље** садржи информације специфичне за протокол као што су MAC адреса извора, MAC адреса одредишта, тип оквира, величина оквира итд. **Корисни терет** садржи IP заглавље, TCP заглавље и саме податке садржаја [7]. Корисни терет пакета треба шифровати XOR-овањем са кључем који генерише алгоритам RC4. Најпре се израчунава контролна сума пакета користећи **полином CRC** (*cyclic*

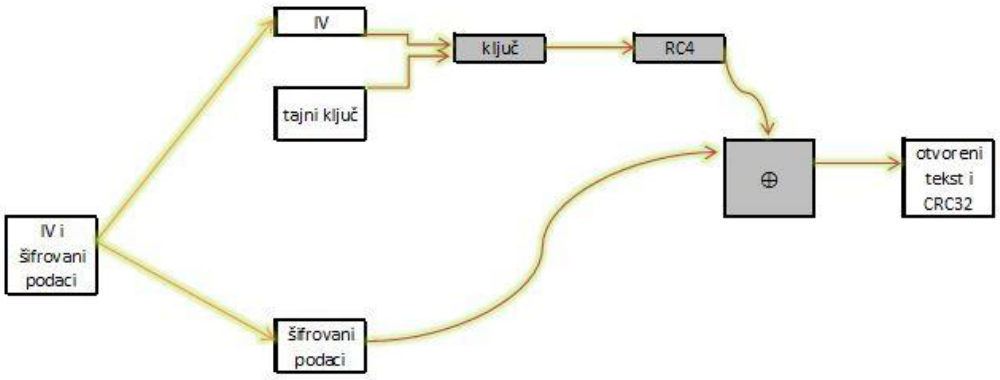
redundancy check) и добијени резултат надовезује на корисни терет. CRC вредност се записује у пољу ширине 4 бајта. Отворени текст је сачињен од корисног терета и контролне суме и може се шифровати. Иницијални вектор, који је коришћен за шифровање пакета, се шаље заједно са шифрованим текстом, али се он шаље нешифровано.

Поступак креирања корисног терета пре слања кроз мрежу је описан на слици 1.



Слика 1. опис поступка шифровања пакета

По пријему пакета, особа **B** издваја иницијални вектор како би на њега надовезала тајни кључ и извршила дешифровање примљеног текста. Особа **B** по дешифровању може да провери интегритет података рачунајући контролни збир дешифрованог текста и упоређујући га са дешифрованом контролном сумом. Поступак дешифровања је описан на слици 2.



Слика 2. опис поступка дешифровања пакета

3.2.1. Анализа протокола WEP

Стандард WEP препоручује да се иницијални векотор мења при слању сваког пакета, како би се избегао напад поновљеним кључем који је у наставку детаљније описан. Међутим, како је дужина иницијалног вектора 3 бајта или 24 бита, постоји 2^{24} укупно могућности за различите иницијалне векторе. Након послатих 2^{24} пакета, неки од иницијалних вектора се мора поновити. Како се вредности за IV бирају насумично, вероватноћа коришћења два иста је већа. Може се очекивати да се после 5000 послатих пакета понови иницијални вектор, што се може повезати са рођенданским парадоксом [5]. **Рођедански парадокс** описује идеју да уколико се у једној просторији налази више од 23 особе, вероватноћа да су неке две од њих рођене истог датума (дана и месеца) је већа од 50%. Уколико се $k = \alpha\sqrt{n}$ објеката вади са понављањем из скупа величине n, тада је вероватноћа да су сви извучени објекти различити једнака:

$$\prod_{i=1}^{k-1}\frac{n-i}{n} = \prod_{i=1}^{k-1}\left(1-\frac{i}{n}\right) \approx \prod_{i=1}^{k-1} e^{-\frac{i}{n}} = e^{-\sum_{i=1}^{k-1}\frac{i}{n}} \approx e^{-\frac{k^2}{2n}} = e^{-\frac{\alpha^2}{2}}$$

Дакле, вероватноћа да су две особе рођене истог датума је приближно $1 - e^{-\frac{\alpha^2}{2}}$. У случају да је вероватноћа да се два рођендана поклапају једнака ½ број особа у просторији треба бити већи од $\sqrt{365\ \ln 4}\ \approx 23$ [1]. Аналогно, ако је вероватноћа да се у комуникацији појаве два иницијална векотра са истим вредностима једнака ½ број иницијалних вектора који су већ ухваћени треба бити већи од $\sqrt{256^3\ \ln 4}\ \approx 4822$.

Структура WEP пакета се може видети на слици 3.

Слика 3. структура WEP пакета

Прва три бајта оквира представљају иницијални вектор, IV као отворени текст. У следећем одељку оквира налази се полазна порука, а затим њена контролна сума.

Главни недостаци протокола WEP последица су недостатака алгоритма RC4. Примена једноставне XOR операције и могућност предвиђања излазних вредности функције PRGA са вероватноћом различитом од $\frac{1}{2}$ у неким ситуацијама је оно што је протокол WEP наследио од алгоритма RC4. Још једна мана је величина иницијалног вектора и могућност његовог понављања приликом слања пакета. Наиме, уколико се претпостави да је величина пакета која се шаље путем мреже 1500 бајтова и да је брзина 5 Mbps, можемо да израчунамо да ће се за половину једног дана послати сви пакети које је могуће направити. Дакле, до понављања пакета ће засигурно доћи. У пракси, иницијални вектор се понавља на сваких 5000 пакета што је на неколико минута [6]. Особа која прислушкује комуникацију и не зна тајни кључ може уз мало стрпљења доћи у посед два пакета са истим иницијалним вектором и кључем. XOR-овањем шифрованих текстова добија се XOR отворених текстова. Даље добијену секвенцу битова може да напада на разне начине не би ли реконструисала два отворена текста. Описана идеја се користи у нападу поновљеним кључем, **Казиски** (*Kasiski*) **напад** [9].

Протокол WEP се уводи 1999. године и како не може да пружи сигурност какву су имале жичане мреже, уводи се поступак шифровања података који се користе у комуникацији. Прва анализа стандарда WEP је извршена 2001. године. За њу су заслужни Н. Борисов (*N. Borisov*), И. Голдберг (*I. Goldberg*) и Д. Вагнер (*D. Wagner*) [12]. Испоставља се да WEP не задовољава сигурносне стандарде, што се од 2001.године и потврдило више пута. 2004. године долази до замене WEP-а, протоколом WPA (*Wi-Fi Protected Access*). Један од напада који се може искористити у протоколу WEP јесте напад FMS [3] који се описује у поглављу 4.

3.3. Алгоритам RC4 у оквиру протокола WEP

Као што је већ речено, тајни кључ за алгоритам RC4 је познат и особи која шифрује и шаље поруку и особи која је по пријему дешифрује. Исти тајни кључ се обично користи за шифровање више пакета, мада је препоручљиво повремено га променити. У оквиру протокола WEP тајни кључ се састоји од иницијалног вектора који се изнова генерише при слању сваког следећег пакета. Кључ настаје надовезивањем тајног кључа, који је увек исти, на иницијани вектор који је специфичан само за актуелну поруку.

Приликом слања пакета, поред шифрованог текста, шаље се и иницијални вектор као отворени текст како би се по пријему могло изршити дешифровање. Наведени поступак представља одступање од начина коришћења кључа код алгоритама RC4, тј. део кључа постаје познат не само примаоцу и пошиљаоцу, него свакоме ко пакет пресретне. Протокол WEP одустаје од традиционалне употребе тајног кључа како би се смањила могућност за напад поновљеним кључем. Поред саме поруке, шифрује се и CRC вредност исте. CRC вредност поруке се још назива и контролни збир. Њоме се проверава да ли је при слању поруке дошло до измена, намерних или случајних.

Идеја да сваки пакет који се шаље користи другачији иницијални вектор можда је слична идеји из [1] и има везе и са тим да повећа кредабилност шифре. Коришћењем дела кључа који је јаван и познат свима који дођу у посед пакета, подстакнуто је више људи да постану нападачи. Уколико се много покушаја за откривањем тајног кључа оконча неуспехом, алгоритам улива више поверења.

Процес дешифровања се обавља тако што по пријему поруке, прималац издваја иницијални вектор из пакета, након чега на њега надовезује тајни кључ и генерише исти низ

кључа. Прималац извршава исти поступак као и особа која је поруку послала. По завршетку дешифровања проверава се да ли је процес шифровања успешан. За то се користи CRC вредност која се налази на самом крају шифрованог текста, такође шифрована. По дешифровању поруке, прималац рачуна CRC вредност поруке коју је добио дешифровањем и упоређује је са CRC вредношћу израчунатом на основу поруке. Уколико су ове две вредности идентичне, процес слања и дешифровања се сматра успешним.

4. Напад на алоритам RC4 у оквиру протокола WEP

Као што је већ поменуто, протокол WEP име недостатке који се могу искористити за откривање кључа који алгоритам RC4 користи за шифровање података. С. Флурер (*S. Flufrer*), И. Мантин (*I. Mantin*) и А. Шамир (*A. Shamir*) су 2001. године објавили рад [3] у коме су описали начин откривања кључа на основу статистичких анализа. Напад је добио име по почетним словима њихових презимена, **напад FMS**.

Полазна претпоставка је да нападач зна први бајт отвореног текста пакета који је пресрео. Ексклузивном дисјункцијом познатог бајта отвореног текста и првог бајта шифрованог текста нападач може реконструисати бајт који је добијен као резултат првог позива функције PRGA приликом шифровања.

За потребе напада FMS, аутори су увели класу иницијалних векотра које су означили као **„слабе“ иницијалне векторе**. У наставку овог поглавља наведени појам ће детаљно бити објашњен. Користећи слабости алгоритма RC4, конкретније предвидљивости првог излазног бајта PRGA функције, може се открити (m+1)-и бајт на основу првих m бајтова кључа.

Иницијални вектори су дужине три бајта. За други бајт се наводи услов да нападача занимају само они иницијални вектори код којих овај бајт има вредност 255. Следећи услов о коме треба водити рачуна је да почетни бајт иницијалног вектора има вредност исту као и индекс бајта кључа који се тренутно анализира. Нападач неће узимати у разматрања оне иницијалне векторе који почињу вредношћу 0, 1 или 2 јер су ти бајтови кључа познати и зависе од пакета који се шаље, то су елементи иницијалног вектора. За последњи елемент иницијалног вектора ограничења не постоје.

Поступак генерисања излазног бајта функције PRGA је описан у поглављу 3.2.1.

Пермутација генерисана након фазе KSA се означава са S. Индекси у овој фази носе ознаке i и j.

У функцији PRGA такође долази до промена у пермутацији S, тако да се она након ових промена означава са S^G док се индекси означавају са i^G и j^G. Променљива t ће бити коришћена да означи индекс елемента у S^G пермутацији са кога је изабран излазни бајт z функције PRGA [2].

Променљива r означава итерацију у којој се врше промене вредности неке од претходних променљивих. Зове се још и итератор. За функцију KSA итератор r узима вредности од 1 до N, док су вредности r у функцији PRGA позитивни цели бројеви, редом. Током функције KSA може се приметити да важи веза између индекса и итератора:

$$i_r = r - 1.$$

За функцију PRGA важи релација описана изразом:

$$i_r^G = r \bmod N.$$

Излазна вредност ове функције се тада може уопштити следећим изразом:

$$z_r = S_r^G[t_r].$$

У складу са овом нотацијом, S_N представља пермутацију након завршетка функције KSA.

Ознаке S_0 и j_0 означавају почетну, иницијалну пермутацију и почетну вредност индекса j, респективно пре позива функције KSA. Слично, важи и S_0^G и j_0^G за функцију PRGA. Приметити да S_N и S_0^G представљају исту пермутацију.

У неким ситуацијама, индекси у пермутацији S, j, i, z, t биће коришћени без додатног описа, али ће из контекста бити јасно на шта се односе.

Потребно је увести још и инверзну ознаку S^{-1}.

$$S[t] = z \text{ онда је } S^{-1}[z] = t.$$

$S[t] = z$ означава да се у пермутацији S на позицији t налази бајт z. Израз $S^{-1}[z] = t$ даје индекс елемента пермутације са вредношћу z.

Проширени кључ који се користи у иницијализацији пермутације S означава се K. Дужине је N. $K[x]$ представља елемент, бајт проширеног кључа K на позицији x.

Како је нападачу познат иницијални вектор, он приступа анализи првог бајта тајног кључа. Он се тада понаша као особа **A** док шифрује поруку. Полази од пермутације S {0,1,2,3, ... ,255} и може да реконструише прве три итерације функције KSA. Из њих нападач може да сазна вредности S_1 и S_2 као и j_0, j_1 и j_2. У наредној, трећој итерацији функције KSA познато је следеће

$$i_{x+1} = x$$

$$j_{x+1} = j_x + S_x[x] + K[x].$$

Функција KSA врши замену елемената пермутације на овим позицијама. Након замене, у овој итерацији $S_{x+1}[x]$ добија вредност $S_x[j_{x+1}]$.

Нека важе следеће претпоставке:

$$S_x[1] < x \quad \text{и} \quad S_x[1] + S_x\left[S_x[1]\right] = x.$$

Заједно, ова два услова чине *повољне услове (1).*

У овом моделу, случајни једнако вероватни догађаји су бајтови кључа. Претпоставка је да у моделу бајтови шифрата бивају независни случајни са равномерном расподелом, што онда омогућује израчунавања вероватноћа у даљем тексту.

У случају да индекс j у итерацијама $\{0, 1, \dots, x+1\}$ не узима неку од вредности 1 или $S_x[1]$, а вероватноћа да се ово догоди је $p = \frac{N-2}{N}$, онда је

$$S_{x+1}[1] = S_x[1] \text{ и } S_{x+1}\left[S_{x+1}[1]\right] = S_x\left[S_x[1]\right].$$

Тако је

$$S_{x+1}[1] + S_{x+1}\left[S_{x+1}[1]\right] = x \text{ са вероватноћом } p = \frac{N-2}{N}. \quad (2)$$

У првом позиву функције PRGA индекс i_r узима вредности из скупа

$$\{x+1, x+2, \dots, N-1, 0\}$$

те у тим итерацијама индекс i „заобилази“ 1, $S_{x+1}[1] = S_x[1] < x$ и $x = S_{x+1}[1] + S_{x+1}\left[S_{x+1}[1]\right]$ у пермутацији S.

Вероватноћа да j у итерацији r неће добити вредности ова три индекса је $\left(\frac{N-3}{N}\right)^{N-x}$, што даље значи да први позив функције, са наведеном вероватноћом, неће променити вредности елемената пермутације S на позицијама једнаким наведеним индексима.

Узимајући у обзир претходно написано, први излазни бајт функције PRGA је описан на следећи начин:

$$
\begin{aligned}
z_1 = \; & S_1^G\left[S_1^G[i_1^G] + S_1^G[j_1^G]\right] \\
= \; & S_1^G\left[S_1^G[1] + S_1^G[S_N[1]]\right] \qquad \text{како је } i_1^G = 1 \text{ и } j_1^G = S_N[1] \\
= \; & S_1^G\left[S_N[1] + S_N[S_N[1]]\right] \quad \text{због напомене да } S_1^G = S_N \\
\\
= \; & S_{x+1}\left[S_{x+1}[1] + S_{x+1}[S_{x+1}[1]]\right], \qquad \text{са вероватноћом } \left(\frac{N-3}{N}\right)^{N-x} \\
= \; & S_{x+1}[x], \qquad \text{са вероватноћом } \frac{N-2}{N} \\
\\
= \; & S_x[j_{x+1}] \\
= \; & S_x[j_x + S_x[x] + K[x]]
\end{aligned}
$$

Уколико су испуњени услови (1), бајт кључа са индексом x се може представити изразом:

$$K[x] = S_x^{-1}[z_1] - j_x - S_x[x] \quad (3)$$

са вероватноћом $\frac{N-2}{N}\left(\frac{N-3}{N}\right)^{N-x}$. За вредности $x > 3$ и $N = 256$ вероватноћа је већа од 0,05.

Нападач прикупља иницијалне векторе и за сваког од њих први излазни бајт функције PRGA. Потом, издваја само оне иницијалне векторе који задовољавају услове (1). Иницијални вектори који испуњавају услове (1) се означавају још и као „корисни“ иницијални вектори. За сваки издвојени иницијални вектор, који називамо корисним, нападач налази кандидата за $K[x]$, користећи израз (3). Ово није коначни избор за бајта $K[x]$, већ само кандидат. Треба увести низ бројача дужине 256 за рачунање фреквенција кандидата $K[x]$. Када се обраде сви иницијални вектори који могу да се искористе за рачунање $K[x]$, најучесталији кандидат се проглашава за највероватнији бајт кључа.

Поступак се аналогно понавља и за све остале бајтове тајног кључа. Када су познати сви бајтови тајног кључа, приступа се верификацији истог. Провера се врши над неколико следећих иницијалних вектора. Уколико се установи да неки бајт кључа није добро изабран, треба покушати са коришћењем следећег по фреквенцији бајта из низа бројача за тај индекс кључа.

У пракси, нападачу је потребно око 5 000 000 пакета како би напад реализовао са вероватноћом већом од 50%. Укупно могућности за појављивање различитих иницијалних вектора има 256^3, како је дужина векотра 3 и за сваки елемент у обзир долази нека од вредности из скупа $\{0, 1, \dots, 255\}$. Нападач може да употреби само иницијалне векторе који за прву координату имају вредност из скупа $\{3, 4, 5, 6, 7\}$, док друга координата има вредност 255. Дакле, нападачу могу да користе само $5 \cdot 255$ иницијалних вектора. Како се тражи да вероватноћа успеха напада буде већа од 50%, претпоставити да је нападачу довољно $5 \cdot 150$ иницијалних векотра. Вероватоћа да је нападач дошао у посед једног од иницијалних векотра које може да употреби јесте $\frac{5}{256^2}$, што значи да би нападач прибележио и анализирао 5 пакета, мора да ухвати 256^2 других. Лако је израчунати да је нападачу потребно $5 \cdot 150 \cdot 256 \cdot 256 \approx 5\,000\,000$ пакета. У поглављу са резултатима и њиховим анализама ове статистике ће бити детаљније показане.

4.1. Пример реализације напада

Описани напад може се илустровати следећим примером.

Како би примери пермутација били поједностављени, у овом примеру за n је узета вредност 3, те је $N = 8$. Сва рачунања се дакле, обављају по модулу 8.

Нека је иницијални вектор којим је шифровање пакета извршено (3, 7, 2) и нека је тајни кључ који је коришћен (2, 7, 5).

КЉУЧ					
IV			ТАЈНИ КЉУЧ		
3	7	2	2	7	5

Поступак генерисања пермутације S, и вредности бројача i и j приказан је кроз наредну табелу.

пермутација S								i	j
0	1	2	3	4	5	6	7	0	3
3	1	2	0	4	5	6	7	1	3
3	0	2	1	4	5	6	7	2	7
3	0	7	1	4	5	6	2	3	0
3	0	1	7	4	5	6	2	4	5
3	0	1	7	5	4	6	2	5	6
3	0	1	7	5	6	4	2	6	5
3	0	1	7	5	4	6	2	7	6
3	0	1	7	5	4	2	6		

Пошто је направљена коначна пермутација S, у последњем реду табеле, остаје да се израчуна излазни бајт функције PRGA.

Излазни бајт функције PRGA је елемент пермутације S на позицији $t = S[i] + S[j]$. Другим речима, резултат ове функције је $S[S[i] + S[j]]$. Подразумева се да се сабирање врши по модулу N.

$i = 1$;

$j = S[1]$;

$Zameni\ mesta\ \left(S[1], S\left[S[1]\right]\right)$;

$t = S[1] + S\left[S[1]\right]$;

$z = S[t]$;

Излазни бајт функције PRGA је на позицији $t = S[1] + S[3] = 0 + 3 = 3$, тј. $z = S[3] = 7$.

Нападач који посматра комуникацију и прикупља пакете може да из пакета сазна две вредности, IV и први излазни бајт функције PRGA. Сада је у могућности да изведе само прва три корака алгоритма KSA. Резултати које нападач добија биће приказани у табели.

КЉУЧ					
IV			ТАЈНИ КЉУЧ		
3	7	2	?	?	?

пермутација S							i	j	
0	1	2	3	4	5	6	7	0	3
3	1	2	0	4	5	6	7	1	3
3	0	2	1	4	5	6	7	2	7
3	0	7	1	4	5	6	2	3	7+1+K[3]=j

У четвртом кораку нападач не може да израчуна вредност променљиве j. За ово израчунавање му је потребан трећи бајт кључа, а то откривање је управо нападачев циљ. Нападач сада може да претпостави да је вредност променљиве j позната. Након одређивања индекса j, извесно је да мора доћи до замене елемената у пермутацији S на позицијама i и j, што значи да елемент са вредношћу 1 неће остати на својој тренутној позицији већ ће заменити вредност са неким непознатим елементом. Нападач сада претпоставља да уколико се до краја функције KSA не буде приступало вредностима на позицијама 0, 1, 3 први бајт тајног кључа ће бити управо први излазни бајт функције PRGA. Последица протокола WEP јесте да нападач зна први бајт отвореног текста пакета који се

шаље, нпр. (AA) и XOR-овањем са првим бајтом шифрованог текста рачуна први бајт фунцкије PRGA. Нападачу је сада позната прва вредност функције PRGA, тако да он зна са којим је елементом морао бити замењен елемент на позицији 3, не би ли његова излазна вредност могла да се добије. Након тога израз $j = 7 + S[3] + K[3]$ у коме су непознате биле две променљиве постаје једначина са једном непознатом и бајт кључа K[3] може да се лако израчуна.

$$K[3] = j + 7 + S[3]$$

Нападач зна да је излаз из функције PRGA вредност 7. У пермутацији коју је он успео да израчуна користећи информације које има о кључу, елемент са том вредношћу се налази на позицији 7. Дакле, $j = 2$. Решавањем претходне једначине добија се да је први бајт тајног кључа број 2.

Уз познавање прве речи тајног кључа, аналогним поступком само са једним кораком више у функцији KSA нападач може да одреди и преостале бајтове тајног кључа. У овом примеру нападач не рачуна статистику за сваки бајт тајног кључа. Пример илуструје метод напада.

5. Програмска реализација напада и добијени резултати

За реализацију напада који је описан у претходном поглављу неопходно је прикупити одређен број пакета које особа **A** шаље особи **B**. Дакле, нападач је у овој фази пасиван и само прикупља пакете. У раду, улазни подаци за напад се не добијају пресретањем, него су излаз посебног програма који генерише потребне податке за напад и бележи их у текстуалну датотеку. Први одељак описује наведени поступак, док наредни детаљно описује поступак напада над прикупљеним подацима. Последњи одељак овог поглавља садржи анализе добијених резултата приликом покретања програма више пута.

5.1. Креирање и прикупљање пакета

Како је за напад потребан и довољан први бајт ове функције, нема потребе радити више од једног позива функције PRGA. Ово би имало смисла када би се у датотеци уз све наведено чувао и шифровани текст издвојен из пресретнутих пакета.

За напад су потребани први излазни бајт из функције PRGA и иницијални вектор, за сваки пакет који се у току комуникације пошаље. Структура датотетеке која се користи у нападу је дата на слици 4 у наставку.

4	ff	12	83
7	ff	c8	7
5	ff	bb	42
3	ff	a6	2e
6	ff	3	ac

Слика 4. структура датотеке са подацима за напад

Прва три броја, раздвојена размаком, представљају иницијални вектор посебно генерисан за одговарајући пакет. Након тога, табулатором је од иницијалног вектора раздвојен први излазни бајт функције PRGA. Сви подаци у датотеци се налазе у хексадецималном облику. У датотеци нема записа иницијалних вектора који не испуњавају услове (1), а то су они вектори којима први елемент иницијалног вектора има вредност већу од дужине кључа и други елемент нема вредност 255 (ff).

Након одређивања иницијалног вектора, додавањем тајног кључа настаје кључ који треба проширити, периодичним понављањем кључа до дужине N. Када је кључ генерисан, приступа се одређивању пермутације S. Поступак је описан у поглављу алгоритам RC4, одељак фаза KSA.

Последњи корак је одређивање и упис првог излазног бајта функције PRGA у датотеку.

Број пакета, тј. редова описане текстуалне датотеке није унапред одређен већ је остављен као параметар који се уноси при сваком покретању програма за шифровање.

Наредни псеудокод приказује начин добијања ове датотеке.

Оператор % означава операцију **mod**, тј. остатак при дељењу.

Ulazni parametri: $BrPaketa, tajni\ ključ\ k$

Izlazne vrednonosti: $IV\ i\ odgovarajući\ keystream\ upisani\ u\ datoteku$

for $i \leftarrow 0$ ***to*** $BrPaketa$ ***do***

IV[0] = (rand() % keyLen) + 3;

IV[1] = 255;

IV[2] = rand() % N;

for $i \leftarrow 0$ ***to*** 3 ***do***

$key[i] = IV[i]$;

for $i \leftarrow 0$ ***to*** $dužinaKljuča$ ***do***

$key[i + 3] = k[i]$;

for $i \leftarrow 0$ ***to*** N ***do***

$K[i] = key[i \% duzinaKey]$;

$keystream = RC4(IV, K)$;

$upiši\ u\ datoteku\ IV, keystream$;

Poštују се услови за први и други бајт иницијалног вектора, док за трећи бајт нема посебних ограничења (осим да мора бити из скупа $\{0, 1, 2, \dots, N-1\}$).

Датотека је сада попуњена и спремна за коришћење у нападу.

5.2. Напад на прикупљене пакете

Нападач као улазне податке користи описану датотеку. У датотеци се налазе само иницијални вектори који испуњавају услов (1) произвољно генерисани. Могуће је да се у датотеци исти IV појави више пута. У том случају је иста и излазна вредност функције PRGA. Такви редови се не узимају у разматрање више од једном. У супротном, долази до тога да се повећа вероватноћа кандидата за кључ добијеног у анализи истог иницијалног вектора више пута. Због тога нападач треба да чува податке о векторима које анализира. Нека се они чувају у низу дужине N, који се при почетку анализе новог бајта кључа иницијализује на нуле. Поред овог, нападачу је потребан још један низ исте дужине у коме ће складиштити податке о томе колико пута се који бајт појавио као кандидат за коначни бајт тајног кључа.

Уколико утврди да иницијални вектор може да искористи за анализу бајта тајног кључа који тренутно разматра, нападач приступа анализи и бележењу добијених резултата. Нека је са A обележен бајт тајног кључа који се открива. Уколико иницијални вектор почиње вредношћу A+3, након чега следи ‘ff’, обрађује се на следећи начин:

Позива се функција KSA, али не N пута већ онолико пута колико је нападачу познато бајтова кључа у том тренутку. Уколико се открива почетни бајт тајног кључа, позната су само прва три бајта кључа, тј. иницијални вектор, те функција KSA има само три итерације. У општем случају, број итерација ове функције је A+3.

Након позива функције KSA може се одредити кандидат за бајт тајног кључа, примењујући добијени израз $K[x] = S_x^{-1}[z_1] - j_x - S_x[x]$. Пошто се испитају сви иницијални вектори који почињу елементом A+3, анализира се низ у који су бележена појављивања кандидата за бајт тајног кључа. Онај бајт који је био најучесталији се узима за бајт тајног кључа на одређеној позицији.

Ulazni parametri: *Datoteka sa podacima za napad, BrPaketa u datoteci*

Izlazne vrednonosti: *Najverovatniji bajtovi tajnog ključa*

```
for A ← 0 to BrPaketa do
    datoteka → IV, keystream
    if IV jeste slab:
        KSA_Xiteracije(A + 3);
        if(j < 2)
            // Narusavamo jedan od prva dva bajta niza S;
              niz_za_narusavanje_prvih_bajtova[A] + +;
        else
            nadjen_j = j;
            nadjen_S = S[A + 3];
            keybyte = keystream − nadjen_j − nadjen_S;
            results[keybyte] + +;
```

5.3. Анализа добијених резултата

Улазни параметар за описани програм за напад је број пакета који су прикупљени и употребљиви за сам напад. Како је позната дужина тајног кључа који треба открити, што је 5 бајтова, може се лако израчунати да максималан број различитих пакета који се могу укључити у анализе јесте 1280. До ове цифре се једноставно долази ако се зна да прва координата иницијалног вектора треба бити вредност из скупа $\{3, 4, 5, 6, 7\}$ друга вредност је константна и износи 255, док трећа и последња координата нема ограничење и за њу постоји 256 могућности.

У идеалном случају, уколико су нападачу познати свих 1280 различитих иницијалних вектора, са великом вероватноћом он може открити бајтове тајног кључа.

У наставку следи анализа резултата добијених програмом за напад са различитим бројем прослеђених пакета који се могу користити у нападу.

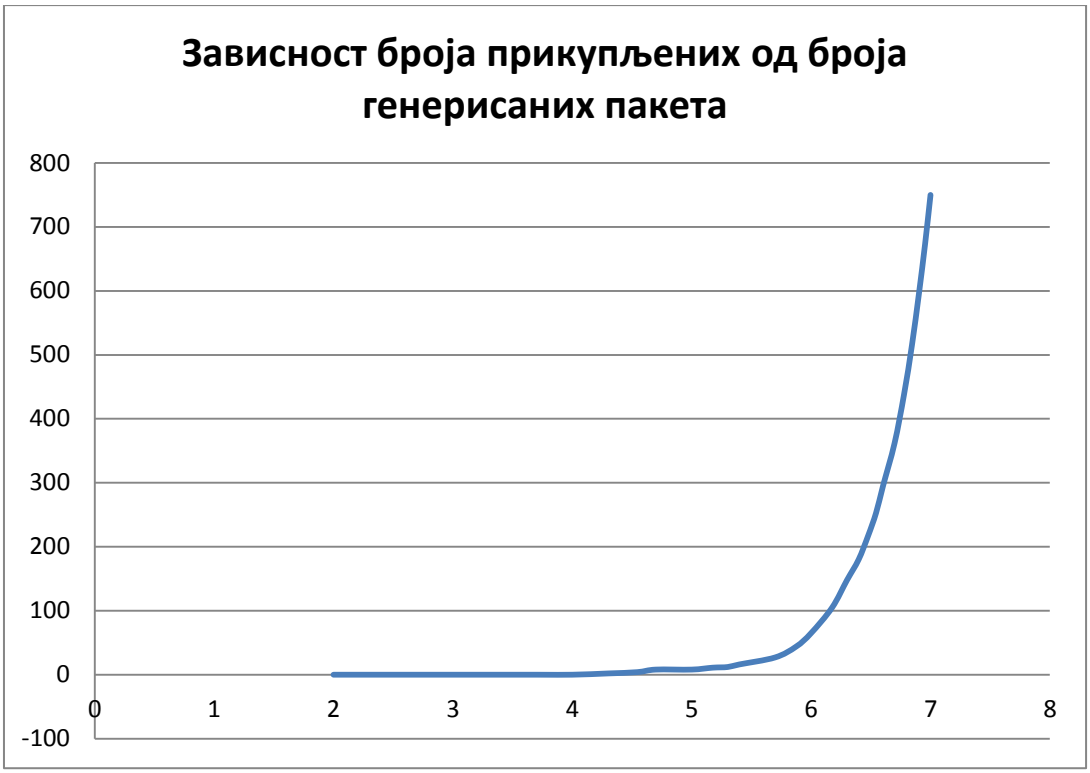
Прва анализа има за циљ да покаже колико је пакета заиста потребно да се генерише и прикупи не би ли се добио довољан број иницијалних вектора који задовољавају услов (1).

Програм за шифровање и креирање пакета је покретан више пута са различитим улазним параметром који представља број пакета који се треба генерисати. Излазна вредност програма је, поред текстуалне датотекте са забележеним иницијалним векторима и број добијених „корисних“ пакета. Добијени резултати су забележени у табели 1.

Табела 1. Удео „корисних“ пакета у зависности од прикупљених

број пакета	број корисних пакета	удео (%) „корисних“ пакета
100	0	0
200	0	0
500	0	0
700	0	0
1000	0	0
5000	0	0
10000	0	0
20000	2	0,010
35000	4	0,011
50000	8	0,016
100000	8	0,008
150000	11	0,007
200000	12	0,006
250000	16	0,006
500000	27	0,005
750000	44	0,006
1000000	65	0,007
1500000	105	0,007
2000000	148	0,007
2500000	180	0,007
3000000	217	0,007
3500000	253	0,007
4000000	295	0,007
5000000	362	0,007
6000000	438	0,007
7000000	516	0,007
8000000	596	0,007
9000000	673	0,008
10000000	750	0,008

Зависност броја пакета који испуњавају услов (1) од укупног броја генерисаних и прикупљених је дата и на графику.



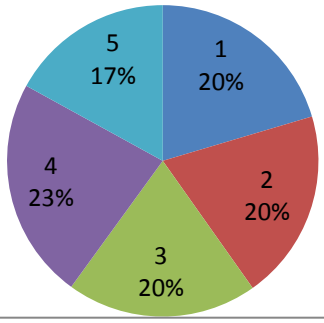
Следећа анализа је показала да број иницијалних вектора који могу да се користе у анализама није једнако распоређен на све бајтове кључа. Може се рећи да је расподела пакета на бајтове није равномерна. На пример, у анализи прикупљених 50 пакета примећено је да је иницијалних вектора са почетном вредношћу 3 било 7, док је број иницијалних вектора који су почињали цифром 4 било два пута више. Детаљнији опис је дат кроз табелу 2 и график.

Табела 2. Расподела иницијалних вектора у зависности од почетног елемента

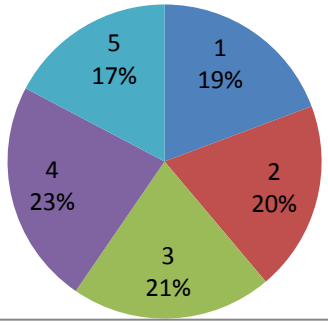
број пакета	IV[0]=3	IV[0]=4	IV[0]=5	IV[0]=6	IV[0]=7
50	7	14	11	9	9
75	12	21	17	12	13
100	17	25	22	21	15
200	33	42	43	45	37
500	102	99	99	115	85
700	135	137	145	162	121
850	163	174	175	191	147
1000	198	207	203	225	167

Покретањем програма за генерисање пакета за напад, као резултат се добија и проценат иницијаних вектора који могу да се искористе за анализу појединачних бајтова. Сразмерно расту броја генерисаних пакета, повећава се и број пакета који су одређени за анализу појединачних бајтова кључа. Процентуални однос пакета за напад је дата на графицима.

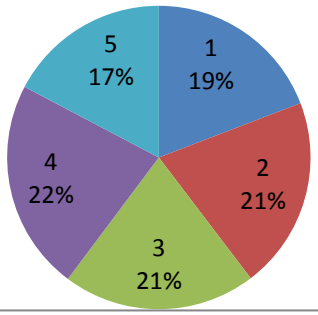
Процентуални однос 500 пакета у односу на бајт кључа који се анализира



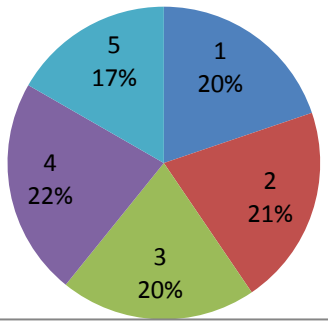
Процентуални однос 700 пакета у односу на бајт кључа који се анализира



Процентуални однос 850 пакета у односу на бајт кључа који се анализира



Процентуални однос 1000 пакета у односу на бајт кључа који се анализира

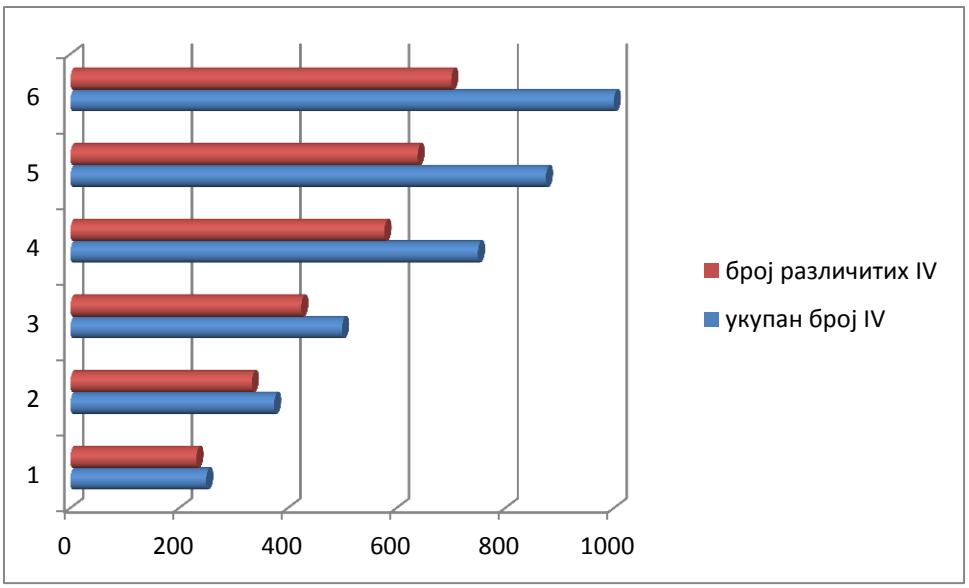


Када нападач прикупља само пакете које може искористити у нападу, треба водити рачуна и о томе да не чува два пута исти иницијални вектор, тј. да избегне дуплирања иницијалних вектора.

При покретању програма за шифровање и генерисање пакета овај услов није разматран. О њему се стара програм за напад. Програми су покретани више пута и као резултат је бележен број прикупљених пакета који ће се користити у нападу и број пакета у којим нема понављања истих иницијалних вектора. Резултати су обрађени кроз табелу 3 и график.

Табела 3. Однос броја прикупљених и броја различитих иницијалних вектора

број IV	**број различитих IV**
250	232
375	334
500	425
750	578
875	639
1000	701

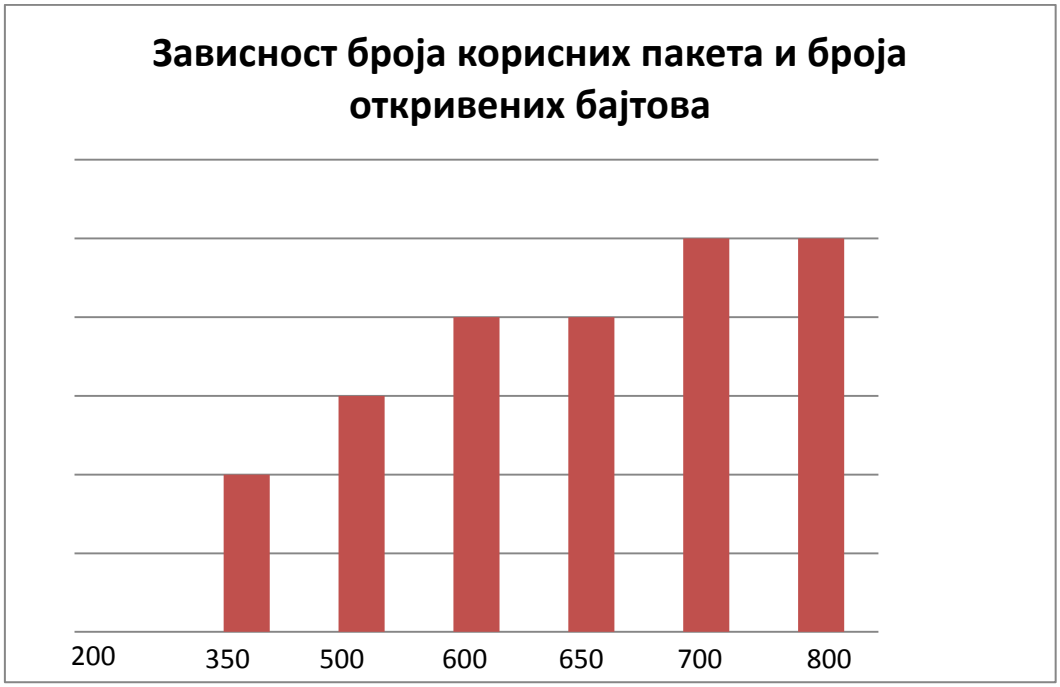


Апсциса одређује број пакета, ордината редни број покретања програма.

Примећено је и да се у зависности од броја „корисних“ иницијалних вектора може открити различити број бајтова тајног кључа. Посвећена је пажња и овој анализи, не би ли се уочила релација између броја анализираних пакета и броја успешно откривених бајтова кључа. Добијени резултати су приказани у табели 4.

Табела 4. Зависност броја корисних пакета и броја откривених бајтова

број пакета	1	2	3	4	5	укупно
200	-	-	-	-	-	0
350	+	+	-	-	-	2
500	+	+	+	-	-	3
600	+	+	+	+	-	4
650	+	+	+	+	-	4
700	+	+	+	+	+	5
800	+	+	+	+	+	5



Током анализе напада, у поглављу 4.1 наведено је да се уколико се испостави да је индекс елемента са којим треба урадити замену у X-тој итерацији функције KSA мањи од 2 тада се иницијални вектор не узима у разматрање и не бележи се као кандидат бајт који би у том случају био добијен. У зависности од броја пакета који се узима у разматрање може се догодити да се за неки бајт кључа не догоди разматрање иницијаног вектора, док се у неким случајевима може не разматрати и више вектора.

Табела 5 показује однос броја укупних пакета и броја неразматраних, по итерацијама у којима су откривани различити бајтови тајног кључа.

Табела 5. Број неразматраних корисних иницијалних вектора по итерацијама

број пакета	**Итерација у којој се откривао**				
	1. бајт	2. бајт	3. бајт	4. бајт	5. бајт
50	0	0	0	0	0
250	0	0	0	0	2
500	0	0	1	0	0
750	0	0	2	1	1
1000	1	2	0	1	1
1500	1	1	2	2	1

Све наведено треба узети у обзир приликом покушаја напада. Задатак нападача није једноставан, али поштујући описану логику напада и уз довољно стрпљења да прикупи довољан број пакета који му могу користити, може открити тајни кључ.

6. Закључак

У овом раду су наведени и објашњени основни појмови криптографије. Дат је опис шифроване комуникације. Описан је алгоритам RC4 и протокол WEP. Пажња је највише усмерена на појам напада. Постоји велики број различитих напада на протокол WEP који су се и практично показали успешним. Кроз пример напада FMS који је у овом раду практично реализован показано је да стандард WEP не задовољава сигурносне стандарде. Након анализе описаног, аутор је практично имплементирао напад и на основу добијених резултата извео статистике описане у поглављу 5.

У будућности, програм коришћен у овом раду би могао бити модификован тако да за напад не користи само описану класу иницијалних векотра који задовољавају услов (1), већ све иницијалне векторе у чији посед дође. Нападачу би много мање времена било потребно за откривање тајног кључа уколико би у разматрање узимао све иницијалне векторе на које наиђе приликом преслушкивања комуникације.

7. Литература

[1] М. Живковић, „Криптографија“, Математички факултет, Београд, 2012

[2] G. Paul, S. Maitra, RC4 Stream Cipher and its Variants, CRC Press, 2012

[3] S. Flufrer, I. Mantin, A. Shamir, Weaknesses in the Key Scheduling Algorithm of RC4. SAC 2001, pages 1-24, vol 2259, Lecture Notes in Computer Science, Springer

[4] IEEE Standard for Information technology— Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE, 2012.

[5] A. S. Tanenbaum, Računarske mreže, Mikroknjiga, 2005

[6] W. A. Arbaugh, N. Shankar, Y.C. J. Wan. You're 802.11 Wireless Network has No Clothes, Wireless Communications, IEEE Volume:9 , Issue: 6

[7] Д. Степановић, Г. Прлина, „Сигурносни пропусти WEP протокола“, INFOTEH-JAHORINA Vol. 9, Ref. F-5, p. 1012-1016, 2010

[8] A. Klein, „Attacks on the RC4 stream cipher Designs, Codes and Cryptography (2008)

[9] http://en.wikipedia.org/wiki/Kasiski_examination

[10] http://sr.wikipedia.org/sr/Zašitita_računarskih_mreža

[11] http://sr.wikipedia.org/sr/Бежична_приступна_тачка

[12] N. Borisov, I. Goldberg, D. Wagner „Intercepting mobile communications: the insecurity of 802.11.“, MOBICOM, 2001

[13] http://sr.wikipedia.org/sr/Mre%C5%BEni_protokol