Прва домашна задача по информациска безбедност

Марија Трпчевска

196002

Документација за имплементацијата на CCMP протоколот

1. Иницијални подготовки

За изработка на проектот се користи програмскиот јазик Java. Податоците и форматот на рамката се преземени користејќи Wireshark packet sniffer, но податочното ниво го користи Ethernet стандардот бидејќи заглавија на рамки кои го користат 802.11 стандардот се преведени од хардверот/драјверот во Ethernet еквиваленти, не прикажувајќи ги вистинските информации. Слично разочарувачки резултати се добиваат и при веб-пребарување за примери од captures кодирани во 802.11i формат. Затоа вредностите потребни при пресметувањата кои се однесуваат на полиња од податочното заглавие се симулирани и/или хард-кодирани.

Решението се состои од шест класи: ClearTextFrame за делење на рамката на заглавие и податоци и нивно понатамошно сегментирање на блокови, AES која беше дадена, CBC за пресметувањето на MIC, CTR за шифрирање на податоците, EncryptedTextFrame за верификација на MIC-от и Main каде што се извршува целосната логика.

1. ClearTextFrame

Конструкторот на класата ја прима рамката во форма на стринг, чии први четиринаесет карактери се претвораат во низа од бајти header (како што беше напоменато, се разгледува Ethernet стандардот кој има 14-бајтно заглавие; на сличен начин би се изведувало одделувањето и за 802.11i, нагодено за големината), а останатите се третираат како податоци. Истите се даваат како аргументи на функцијата Separate\_in\_128\_bit\_blocks(byte[] array) која низата од бајти ја претвора во матрица со 16 колони, така што секој ред ќе соодвестува на еден 128-битен блок. Ако елементи од последниот блок ако не се иницијализирани експлицитно, тие имаат вредност нула согласно конвенциите на Java, што воедно игра улога и на 0-byte padding. Дополнитело, print\_packet\_in\_blocks() служи за верификација на точната поделба на стандарден излез.

1. Main (прв дел)

Како аргумент во конструкторот на објектот frame од класата ClearTextFrame се дава следниот пакет:

dp...st.:.T...E.....@.)....w.....e.P..S...u'qYP.......HTTP/1.1 200 OK..Date: Sun, 31 Oct 2021 15:47:41 GMT..Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.25 mod\_perl/2.0.11 Perl/v5.16.3..Last-Modified: Sun, 31 Oct 2021 05:59:02 GMT..ETag: \"80-5cf9fc1d3fb42\"..Accept-Ranges: bytes..Content-Length: 128..Keep-Alive: timeout=5, max=100..Connection: Keep-Alive..Content-Type: text/html; charset=UTF-8....<html>.Congratulations. You've downloaded the file.http://gaia.cs. umass.edu/wireshark-labs/HTTP-wireshark-file1.html!.</html>.

Истиот се дели на блокови и печати. Се преминува кон креирање на nonce-от; вредноста на полето Package Number (PN) во заглавието се симулира преку 6 бајтна низа од случајни вредности, MAC-от на праќачот е хард-кодиран бидејќи е достапен како поле и во Ethernet (а неговиот хексадецимален стринг се претвора во низа од бајти преку public static byte[]hexStringToByteArray(String s)). Полето QoS за квалитет на сервис е хард-кодиран со 0x28 бидејќи e една од можните вредности кои може да ги има во реалноста, а се однесува на QoS Data поттип (не сите пермутации од 8 бита се дозволени како вредност за да се програмира како случајна 8-битна низа). Така, спојувањето на PN-MAC-QoS во тој редослед го дава nonce-от кој би имал 6+6+1= 13 бајти, или 104 бита.

Како привремен клуч се користи хард-кодиран стринг “OVERLORD” и се става како клуч преку AES.setKey(String myKey) за сите понатамошни шифрирања. Се креира инстанца cbc\_mode\_encrypt од класата CBC и се праќа блоковски разделеното заглавие и податоци, nonce-от и клучот во конструкторот.

1. CBC

Иницијализирачкиот вектор (IV) се прави така што се доделуваат 16 бајти (128 бита) првично и се копираат 104-те на nonce-от од нултата позиција, добивајќи три 0-иницијализирани вредности на крај како padding. Главната функција е calculate\_MIC() која го шифрира IV -то со AES користејќи го клучот. Во низата бајти cipher\_block се чуваат сите меѓурезултатни шифрирани блокови, започнувајќи со првиот блок на заглавието кое е комбинирано преку функцијата XOR (за извршување исклучиво или врз две низи) со шифрираниот IV. Понатаму низ циклус се минуваат преостанатите блокови од заглавието и податоците, внимавајќи како IV да се користи претходната вредност чувана во cipher\_block. Последната од нив пак го креира MAC-от (Мessage authentication code) за рамката. Во променливата MAC се чуваат најзначајните 64 бита од 128-битниот cipher\_block, а пак 128-битниот MIC е создаден од копирање на MAC и 64 битен 0-padding; Ctr\_Preload е копија на nonce со 24 битен 0-padding. Истиот се шифрира и се врши XOR со MIC за да се добие конечниот шифриран MIC како повратна вредност. Резултатот се печати на стандарден излез.

1. CTR

Во Main се креира инстанца ctr\_mode\_encrypt на класата CTR и во конструкторот се даваат како аргументи блоковски разделениот податочен дел, nonce-от и клучот и се повикува функцијата encrypt\_payload(). Се забележува try-catch блок кој фрла исклучок од класата FailedEncryptionDecryption секогаш кога не се доставени валидни вредности за аргументите. Во нормален тек, се креира променлива Ctr\_Preload која е добиена со спојување на nonce со 24-битен одбројувач counter иницијализиран на 1, се шифрира, се врши XOR со блокот податоци и се инкрементира одбројувачот во циклус. Во имплементацијата постои помошна низа од бајти counter\_4\_bytes која има 32 бита со цел да може да се преведе во 32 битна integer вредност (како што налага Java), таа да се инкрементира, да се врати повторно во низа бајти и да се земат последните три за нова вредност на counter. Се враќаат шифрираните матрични блокови.

1. Main (втор дел)

Имајќи го MIC-от и шифрираните податоци, истите се користат како аргументи во конструкторот за encrypted\_frame од класата EncryptedTextFrame, симулирајќи испраќање. Се повикува функцијата print\_payload() за да се испечати шифрираниот податочен дел и се преминува кон дешифрирање со гореспоменатите функции. Преку нова инстанца ctr\_mode\_decrypt од CTR се дешифрираат податоците и се печатат. Нова инстанца cbc\_mode\_decrypt од CBC служи за пресметка на MIC-от од дешифрираните податоци и првичното заглавие кое препраќа како аргумент. Конечно, може да се повика функцијата compare\_MICs(byte[] MIC\_new) во try-catch блок која се наоѓа во класата EncryptedTextFrame, а фрла исклучок од класата CompromisedIntegrity ако MIC-от добиен од дешифрираните податоци и заглавието не се совпаѓа со пратениот MIC и прикажува соодветна порака.

1. Извршување на кодот

А) Нормален тек

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |

Б) FailedEncryptionDecryption исклучок предизвикан со null nonce и CompromisedIntegrity предизвикан од различни вредности за изворен и пресметан MIC

|  |  |
| --- | --- |
|  |  |