**HW1 Writeup**

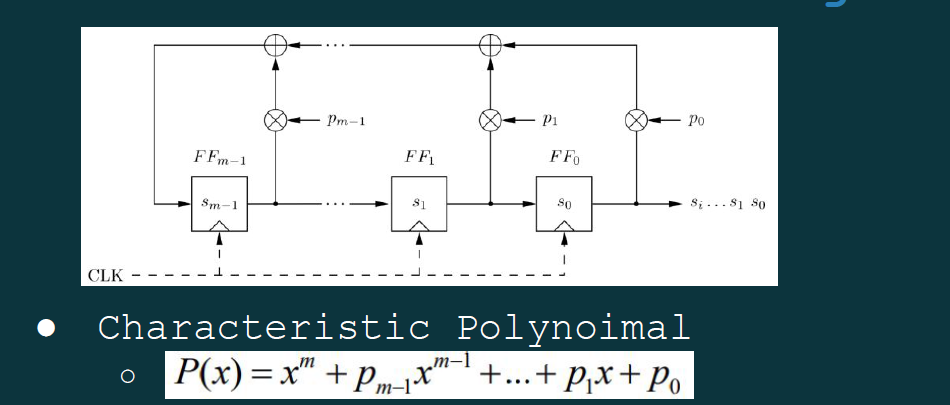
**R10922043 黃政瑋**

**CTF account: cwhuang1937**

1. **nLFSR**
   1. 轉成Characteristic Polynomial

首先觀察step function，可以發現他的過程相似於nonlinear-LFSR，又已知state的初始值是隨機產生的64bits，因此我們可以將這題轉成在**GF(2^64) 上的Characteristic Polynomial**。接著將’1’ + poly reverse來當作GF的modulus。

**modulus的由來:** 之所以reverse是因為GF吃的係數是降冪的形式，而state的bits是升冪的形式，1則是補上x^64項的係數。(這邊跟組員**R10922046李柏漢與R10944011吳添毅**討論非常久，這邊最一開始是參考到網路上某份相似題目的writeup，但對他如何取modulus這部分一直很不明白，我們三個討論了很久才弄懂為何要這樣取)



一張含有 文字 的圖片

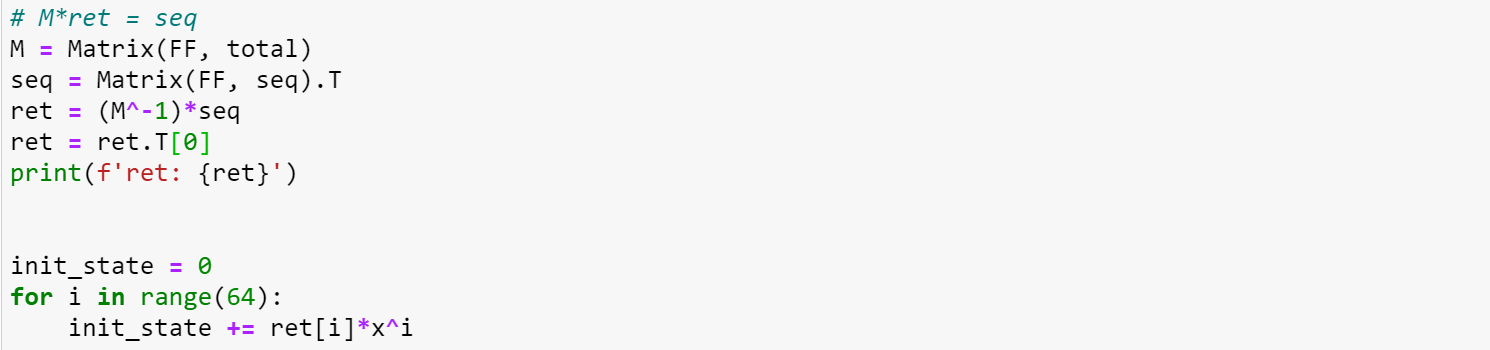
自動產生的描述

* 1. 求出state-transition matrix

由source code可知，每次的random function會經過43次step，也就是state會轉移43次，因此這邊可以列出64條state-transition的式子分別對應初始state做了random function 1~64次，此64條式子形成的係數即為state-transition matrix。

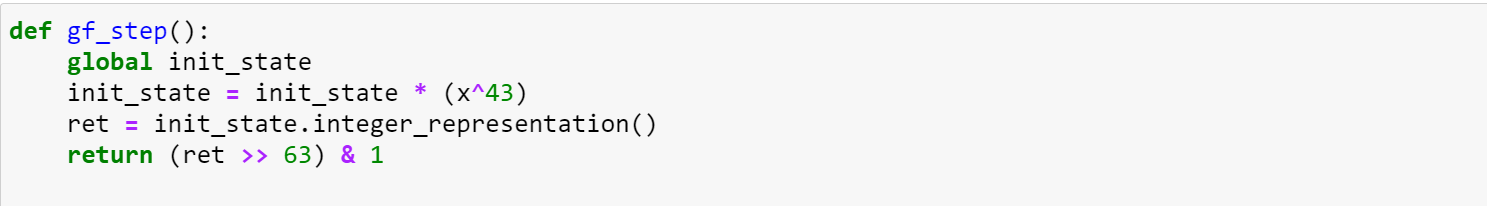
* 1. 求出init\_state

接著用pwntools連上server後，全部故意猜0並根據money的變化來取得前64次的結果，接著用前面的state-transition matrix做矩陣運算後可以得到init\_state。



* 1. 取得flag

取得init\_state後，先做64次自己定義gf\_step(一次gf\_step等同於做了43次step)，來讓init\_state來到我們前面亂猜完64次後的state，接著照gf\_step每次回傳的結果來往下猜，即可100%預測出之後的0、1，最終即可超過2.4元拿到flag。



Reference:

<https://hackmd.io/@cXpZn6ltSku4Vwx_OL0bqA/SyyxioFgI?fbclid=IwAR2rdKeIMoHmXtc3dpou8babWXKhVR4zp2PatfipAHJB9z6lBUT1xjGFcJE#winner-winner-chicken-dinner>

1. **Single**
   1. 求係數a、b

首先這題並不知道橢圓曲線的係數a、b，因此先將題目提供的三點(G、A、B)取其中兩點代入方程式，可以得二元一次聯立方程組。由於這兩個式子都是同餘方程，這邊用**sage的solve()**來解，接著做一些數學運算後即可得到係數a、b。

一張含有 文字 的圖片

自動產生的描述

* 1. 判斷是否為singular curve

將求出來的a、b帶入(4\*a^3 + 27\*b^2)%p，會得到0，即可確定這是個**singular curve**。

* 1. 求出phi function中的alpha、beta

對原式call .**factor()** 後，比對係數即可得到alpha、beta。

一張含有 文字, 裝置, 量表, 儀錶 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

* 1. 由公式回推出dA

由A = G\*dA，可以推得phi(A) = phi(G\*dA) = phi(G)^dA (singular curve中node的性質)，接著帶入**discrete\_log()**即可解出dA。

一張含有 文字 的圖片

自動產生的描述

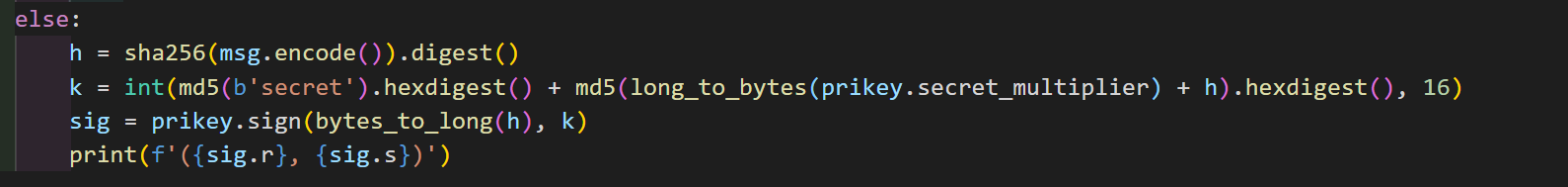
最後取出B\*dA的x，即可以得到key值k，拿k來還原出flag。

Reference:

<https://crypto.stackexchange.com/questions/61302/how-to-solve-this-ecdlp>

1. **HNP-revenge**
   1. 判斷題目

觀察題目所給的k會發現，他是一個256bits所形成的integer，其中**前128bit為一定值**，而後128bit來自一**隨機產生**的prikey與bytes Kuruwa所組成，由此可知，這個h的high bits是已知的，故可套用講義上的lattice basis，但這邊**k的high bits並不是都為0**，故下面求解u、t時需做些代換。



一張含有 文字 的圖片

自動產生的描述

* 1. 取得r1、s1、r2、s1

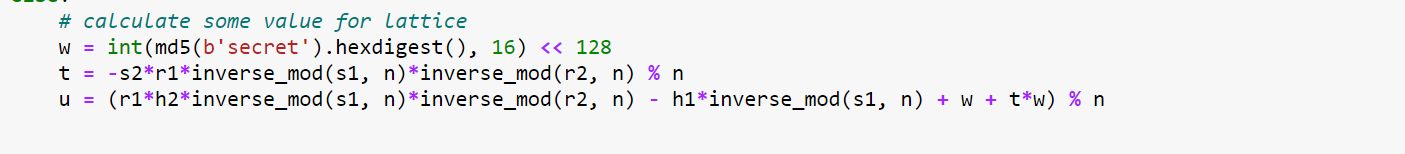
與本週的lab1相同，前兩次option先用不同的兩個message來得到兩組r與s，以便第三次option的計算。

* 1. 計算lattice basis中的u、t

進入第三次option時，首先將k中前128bit的值用一個變數w來替換，接著照著講義的公式移項之後帶入數值，即可求得u、t。

一張含有 文字 的圖片

自動產生的描述

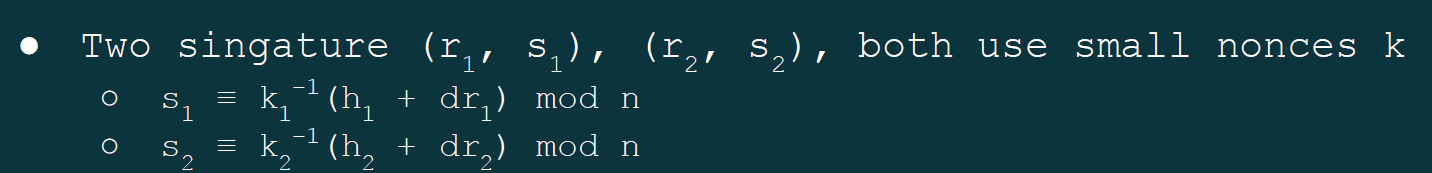


* 1. 用LLL求解approximated SVP

這邊K取用n的開根號-1(也就是2^128-1)，找出符合條件的vector，即abs(k1)和abs(k2)皆小於K。求出來後加上原本的w即可還原出原始的k，接著帶入公式後即可還原出d。

一張含有 文字 的圖片

自動產生的描述



* 1. 由d反推出題目的prikey

用題目固定的G與推出來的d即可產生pubkey、prikey，接著再用該prikey去簽bytes Kuruwa的sha256，即可得到正確對應的r、s，最終即可通過認證並得到flag。

* 1. **解決不能每次成功得到flag的問題**

每次的prikey產生都是random的，因此每次對應到的lattice basis也都不同。**LLL只能保證求出一個近似SVP的vector**，而這vector不一定剛好對應到題目所要的k，會導致之後推導出來的d未必是題目一開始簽所用的d，因此用這個d產生的prikey來簽會得到Bad signature。這邊將pwntools連上server前用個無限迴圈包住，直到求出flag才跳出迴圈，解決無法每次成功求解的問題。