知識天地

一些破密與加密最近的鬥爭

楊柏因（本院資訊科學研究所副研究員）

一般來說，資訊的安全防護被破解或是被侵入系統，很少是因爲密碼學的理論本身造成的，絕大多數是人爲操作上或是程式、系統設計的錯誤被利用。即使如此，相信多數人仍有興趣知道最近學理上種種關於破解密碼的理論。

破密的理論與實際

所謂密碼系統，也就是密碼學所稱的基本操作[primitives]，都有設計者希望達成的安全性。這就是用純暴力的方法算出攻擊者不該有的東西—沒加密的原文，或是和眞品具有相同效果的數位簽章等—其理論上需要的時間。例如說 DES 加密系統的密碼有 56 個位元，所以用蠻力硬試當然可以在加密 2^{56} 次的時間以內算完。以此爲準，密碼學家所謂的破解成功，就只表示找到方法可以把所需要的時間壓低到設計者要求的時間以下。換句話說當設計者希望的安全性是 2^{256} 那所謂破解可能是 2^{200}，2^{100}，甚至 2^{20}。當然其實用性也差距很大。在密碼學研討會的論文發表，經常是這樣做結：「…所以可以用 2^{70} 個位元的的記憶體[這或許需要把地殼中所有的矽拿來做 RAM]和 2^{110} 時間算出[128 位元]密碼，也就是破解完成」—而此刻大家會鼓掌並承認他的確破解成功了。

最近的新型攻擊：代數破密(Algebraic Cryptanalysis)

密碼系統可分爲對稱式(秘密金鑰)和非對稱式(公開金鑰)兩類。後者的特質是：可以在被壞人全程監聽時使用；顯然這很有用，但是同時它也極緩慢，不適合持續操作。因此，公開金鑰密碼系統應用於加解密的主要用途是傳送對稱式系統的當次金鑰(session key)，再以對稱式系統加解密。

對稱式密碼系統分爲兩種：區塊式如 DES、AES，和串流式如 RC4。串流式的一般特徵是比較快，因此初期也廣爲採用。串流式的加解密簡單的來說就是僞亂數生成器(pseudo-random number generator, PRNG)！我們知道，一個亂數生成器，例如大家熟悉的 rand()函數的理想狀態如下：給定不同的種子(seed)，即可輸出一個旁人無從預測，甚至無法和眞正純隨機的數列區隔，但只需要擁有種子即可重現的「亂數」列。

假設我們有一串無限長的而純隨機的位元列[由 0 和 1 組成的序列]，而接收者和發送者已經各預先備妥一份，那麼理論上只需要把要交換的明文也表達成一串位元，並且和這串位元列—我們稱之爲單次使用密碼本(one-time pad)—來作一個位元一個位元的邏輯上 XOR 操作，即成爲理論上無懈可擊的密碼，所以如果我們設想僞亂數生成器的輸出串列—我們稱之爲密碼串列—眞的和純隨機的位元串列無法區隔，那麼一切就很完美了。

很不幸，事實上不可能如此，這類的密碼系統必須從種子—密碼學家通稱爲金鑰(key)—開始，造出一個初始狀態(initial state)，此後藉由維持一些狀態資料(state)而產生一個僞亂數串，以代替眞正的單次使用密碼本。

在程式語言中的 rand()以至於統計學家實作模擬的僞亂數產生器，基本上都根據某個線性遞迴式加上一個輸出函數(output filter）而來，例如 GNU libc 程式庫中的線性 rand()的輸出，是先從種子 $r_0 = s$ 製造出初始狀態：

$$\mathbf{r_i = (16807 * (signed\ int)\ r_{i-1})\ mod\ (2^{31} - 1)\ [i = 1\cdots30],} \quad \mathbf{r_i = r_{i-31}\ [i = 31\cdots33]}$$

然後進行線性遞迴式 $r_i = (r_{i-3} + r_{i-31}) \bmod 2^{32}$ (for $i \geq 34$)，捨棄其前 343 個數字，最後使用簡單的輸出函數 $o_i = r_{i+344} >> 1$(>>1 代表向右移動，即除以二)。所有您可能在寫 C/C++時簡單呼叫到的亂數程式都具有同一形狀。

作爲統計上的工具時，僞亂數列有很多學理上必須滿足的條件，各位可以參考 Don Knuth 聖經(*The Art of Computer Programming*)的第二冊；事實上做模擬，有不少簡單的僞亂數產生器[例如有名的 Mersenne Twister，松本眞和西村拓士所提出，http://en.wikipedia.org/wiki/Mersenne_twister]就足夠良好了，但作爲加解密系統的僞亂數列還

需要更好的性質。早在數十年前，數學家就發現了所謂 Berlekamp-Massey 演算法：任何長度爲 *n* 的線性遞迴式可以由他的前 2n 項唯一且迅速的決定，因此輸出函數必須是非線性的；在 1976 年 Rotenhaus 的引理指出根據線性遞迴系統與某些型式的輸出函數—所謂彎曲的(bent)的函數—所造出來的僞亂數列不容易用線性的方法和眞正隨機的數列區隔。有很長的時間，密碼學家認爲只要選擇適當的線性遞迴式—硬體上稱爲線性回饋的遞移暫存器(LFSR)－和輸出函數，就保證安全。

事情事實上並不這麼簡單，從 2002 年開始，幾乎所有這一類型的密碼系統被破得乾乾淨淨，我們舉一個有點誇張但是事實上發生的例子來說明爲什麼會發生。一個稱之爲 Toyocrypt 的加解密系統幾乎成爲日本的國家標準，以取代老牌的 RC4 系統(前仍用於某些無線網路的加密)。Toyocrypt 使用一個多達 128 個單位元變數的 LFSR，加如下的輸出過濾函數(filter function)：

$$f(s_0,..,s_{127}) = s_{127} + \sum_{i=0} s_i s_{\alpha_i} + s_{10}s_{23}s_{32}s_{42} + s_1s_2s_9s_{12}s_{18}s_{20}s_{23}s_{25}s_{26}s_{28}s_{33}s_{38}s_{41}s_{42}s_{51}s_{53}s_{59} + \prod_{i=0}^{62} s_i,$$

其中$\alpha_0, \ldots, \alpha_{62}$是從｛63, …, 125｝選出來的排列。Toyocrypt 每被呼叫一次，LFSR 就變換一次由 128 個一位元的變量($s_0, \ldots, s_{127}$)構成的狀態，同時算出 f 作爲輸出串列的該位元。剛剛提到的引理說：這樣的密碼很難用傳統攻擊去破密。很不幸的，問題同時也出在這個輸出函數上，因爲我們可以把式子的下半段全部刪掉，成立的機率應該還是很接近 1；事實上，差不多是 $1 - 2^{-17}$。

有了近似成立的方程式，就可以求解了，我們只要蒐集足夠多的資料，就可以把 key 和 state 當成變數，把方程組一次解出來。在 Toyocrypt 的例子，因爲方程式不成立的機率 2^{-17} 太小了，以至於我們可以蒐集 10^5 筆資料，仍然可以很有信心地說方程組會成立。如果運氣不好，只需要多算一兩次。

使用代數手段來破解密碼系統的方法，相對於兩千年以前的密碼學家以統計作爲主要手段的古典攻擊，被統稱爲代數攻擊，其中常常會以解方程式作爲最後一步。數百個變數和方程式，這樣的方程組還能解嘛？眞是不幸，目前的解方程式技巧—特別是當變數僅爲 0 或 1 時—有長足的進步。有興趣的讀者可以看三篇有關 XL 演算法的論文(http://precision.moscito.org/by-publ)。

因此，即使製作人認爲這個密碼的安全性是 $>2^{80}$時間[單位是進行一區塊 3DES 密碼系統加密所需的時間，這是密碼學家常用的一個標準]，審查的專家也同意，但是卻可以用 2^{60}以下時間完成破密。

這告訴我們什麼？很明顯我們應該避免採用這一類型的密碼，可惜有一個已經被標準化了，就是藍芽無線傳輸設備中所使用的密碼系統 E0，所以如果您使用藍芽鍵盤或是耳機，一個持有特製天線的壞人可以在百公尺以外截聽到所有被傳遞的資訊，然後很快的解碼出所有傳遞的資料。如耳機拿來聽聽歌當然沒關係，但是如果可能用來輸送機密，例如鍵盤，那就有點糟糕了。

我們也必須小心的設計將來的系統以免重蹈覆轍。今天的系統設計人很可能會採用區塊式密碼系統：美國國家標準局從 1997 年開始徵求新一代的加密標準，並在 2001 年底開始正式採用應徵者之一爲先進加密系統[AES，可見 http://en.wikipedia.org/wiki/Advanced_Encryption_Standard_process]。在十年的努力研究之後，大家目前仍然相信它在可預見的將來是可靠的。

我們的研究：XL 解方程組的分析

XL 可以說是 Gröbner 基底法的簡化版—原理比較單純，程式的撰寫也比較簡單。假設在基體 K 上，我們有 n 個變數 $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ 的 m 個二次（更高次也類似）方程式 $l_1(\mathbf{x}) = l_2(\mathbf{x}) = \ldots = l_m(\mathbf{x}) = 0$ 且 $n \le m$，那麼 D 次的 XL 操作如下：

i. 第一步驟是 X，X 表示乘法，或是延伸(eXtended)。我們將每個 $D^{\sim}$ 2 或更低次的單項 $x_1^{\alpha 1} x_2^{\alpha 2} \cdots x_n^{\alpha n}$ （通常記作 $\mathbf{x}^{\alpha}$，其中 $\Sigma_i \alpha_i \le D$ - 2）乘以所有方程式 $l_j(\mathbf{x}) = 0$。

ii. 第二步驟是 L，L 表示線性化(linearize)。我們把每個 D 次或更低的單項當成獨立的變數，並把這樣求出的 $m C_n^{n+D-2}$ 條方程式(當 $q = |K| \le D$ 時會變少)做高斯消去法。而變數的排序，也就是各單項式被消去的順序，採用相反字典式(reverse lexicographical order)。那麼最後被消去的，是第一個變數 x_1 的單項。

iii. 如果獨立的方程式至少有 $C_n^{n+D} - D$ 條(這個下限在 $|K| \le D$ 時也會變少)，我們應該已經得到僅含 x_1 單項式的一條以上方程式，消去 x_1 以外的任何變數。接著可以採用 Berlekamp 演算法求出 x_1。如果還有沒做完的變數，那麼重覆上面的動作，得到每一個變數。

這裡理論上的分析非常複雜，需要應用到組合學和複變函數的技巧，我們不多提它，只簡單告訴各位讀者，XL 在理論上不如更複雜的演算法，但是使用它，我們可以應用疏落[sparse]矩陣的技巧—這樣做出來的係數矩陣總是絕大多數爲零—進而以更少的時間和資源求出解。我們發現，解方程式的時間仍是指數成長，但是成長率比原來想像中的要慢許多。當數字 n 大概幾十的時候，n 個方程式解 n 個 GF(256)中的變數只需要大約一個 n 的多項式乘上 $2^{2.4n}$ 的時間！

「理論上安全」與 QUAD(256, 20, 20)的分析和破解

由於這些破密的進展，最近的密碼學界開始探討所謂理論上的安全性，最近的學界有很多文章如此：假設 ZZ 問題不可解，如果密碼系統 AA 可以破解，那麼把這個破解的過程當成黑盒子來使用，我們就可以證明一連串定理說我們可以做到 BB，CC…最後做到 ZZ-足見 AA 是安全的！

我們最近在這方面的研究指出，使用這一類的定理和前面第三節的爭端一樣，需要注意細節，因爲沒有什麼問題眞的是不可解的，頂多只是很難解而已；如果 ZZ 問題可以經過兩次的破解 AA 系統的過程而解出，同時我們認爲要解決 ZZ 問題至少需要 2^{100} 時間，那麼破解 AA 系統就至少需要 2^{99} 時間。這樣的證明大致沒有問題，但是如果定理的證明中 ZZ 問題需要經過 2^{40} 次破解 AA 系統呢？那我們只證明了這個破解 AA 系統需要 2^{60} 時間。這對於現代密碼學家來說是不夠的！

在去年的歐洲密碼學大會中，知名的法國密碼學家 Gilbert 和他的兩位學生提出如下的串流式密碼系統：給定初始狀態 $\mathbf{x_0}$ 是 n 個 K=GF(q)的元素所組成，更新規則 $\mathbf{x_{i+1}=Q(x_i)}$，每次的輸出 $\mathbf{y_i=P(x_i)}$，**P, Q** 都是隨機選出的 K^n 到 K^n 的函數，他們提出一些定理認爲解多元二次方程式組的困難度可以保證這個密碼系統的安全性。而 $q^n = 2^{160}$ 時，可以得到相當快而且安全的效果。很不幸，由於前面提到的盲點，我們發現，只要解

$$\mathbf{y_0=Q(x_0),\ y_1=Q(P(x_0))}$$

的方程組那麼在 q=256，n=20 時，這二次和四次的混合方程組可以用 XL 在 2^{60} 乘法所需的時間以內計算出來，這個研究發表在最近於盧森堡舉行的快速軟體加密研討會(FSE 2007)，並且獲得很好的迴響。目前對於尋求更安全的參數組合和更好的破密方式的後續研究仍在各地進行當中。

小結

這裡所提供的或許只適合在茶餘飯後提供一些樂趣。但是或許足以提醒您密碼學的理論眞的是日新月異，隨時都可能會有新的發展出現。希望聰明如您們能夠領會到資訊安全需要時時掛在心上的重要性。