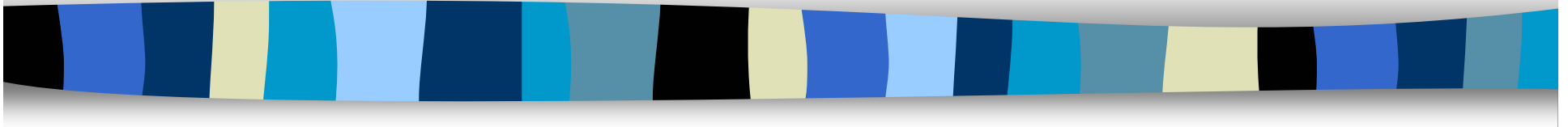


Cryptography CS 555



Review of HW2 and HW3

HW 2: Problem 1

- Consider the following insecure PRNG: Given prime p and a, b, c , the seed is (s_1, s_2) , repeat the following steps:
 - output $((cx_1+x_2) \bmod p)$ when (x_1, x_2) is internal state
 - set new state to $(ax_1+x_2, bx_2+x_1) \bmod p$
- let y_1, y_2 be two outputs
 - $x_1=s_1, x_2=s_2, y_1=cs_1+s_2$
 - $x_1=as_1+s_2, x_2=bs_2+s_1, y_2=c(as_1+s_2)+bs_2+s_1,$
 - $y_2=c(as_1+y_1-cs_1)+b(y_1-cs_1)+s_1$
 - $(ca-c+bc+1)s_1=y_2-cy_1+by_1$

HW2: Problem 1

- What if a, b, c are not known?
 - one may try to use five equations to solve a, b, c, s_1, s_2
 - however, this is very difficult, as things are not linear
 - an elegant solution due to Sarah
 - idea: write $y_{i+2} = A y_{i+1} + B y_i$
 - then using four outputs and $y_3 = A y_1 + B y_2$, $y_4 = A y_3 + B y_2$, one can solve A and B , and predict all future outputs
 - Using $y_1 = c x_1 + x_2$ can eliminate x_1
 - Using $y_2 = c(ax_1 + x_2) + (x_1 + bx_2)$ can write $y_2 = F y_1 + G x_2$
 - Using $y_3 = F y_2 + G (bx_2 + x_1)$ and eliminate x_1 and x_2

HW2: Problem 1

$$y_1 = cx_1 + x_2$$

$$y_2 = c(ax_1 + x_2) + (x_1 + bx_2)$$

$$= acx_1 + cx_2 + x_1 + bx_2$$

$$= a(y_1 - x_2) + (b+c)x_2 + c^{-1}(y_1 - x_2)$$

$$= (a+c^{-1})y_1 + (b+c-a-c^{-1})x_2$$

$$y_3 = (a+c^{-1})y_2 + (b+c-a-c^{-1})(x_1 + bx_2)$$

$$= (a+c^{-1})y_2 + (b+c-a-c^{-1})x_1 + b(b+c-a-c^{-1})x_2$$

$$= (a+c^{-1})y_2 + (b+c)c^{-1}y_1 - c^{-1}y_2 + b(y_2 - (a+c^{-1})y_1)$$

$$= ay_2 + y_1 + bc^{-1}y_1 + by_2 - b(a+c^{-1})y_1$$

$$= (1-ab)y_1 + (a+b)y_2$$

$$cx_1 = (y_1 - x_2) \quad (1)$$

$$x_1 = c^{-1}(y_1 - x_2) \quad (2)$$

$$\begin{aligned} (b+c-a-c^{-1})x_2 \\ = y_2 - (a+c^{-1})y_1 \end{aligned} \quad (3)$$

$$\begin{aligned} (b+c-a-c^{-1})x_1 \\ = (b+c-a-c^{-1})c^{-1}(y_1 - x_2) \\ = (b+c-a-c^{-1})c^{-1}y_1 - \\ (b+c-a-c^{-1})c^{-1}x_2 \\ = (b+c-a-c^{-1})c^{-1}y_1 - \\ c^{-1}(y_2 - (a+c^{-1})y_1) \\ = (b+c)c^{-1}y_1 - c^{-1}y_2 \end{aligned} \quad (4)$$

HW 3: Problem 1

- DES
 - S box Substitution
 - IP, IP⁻¹, P Transposition
 - expansion Substitution
 - XOR One-time pad
- AES
 - SubBytes Substitution
 - ShiftRows Transposition
 - AddRoundKey one-time pad

HW3: Problem 3a

$$h(x_1, y) = E_{x_1}[y] \oplus y$$

$$h(m || x, y) = E_x[h(m, y)] \oplus h(m, y)$$

- Part a: for DES

- $E_{c(K)}[c(x)] = c(E_K[x])$

- given $h(x_1 x_2 \dots x_k, y)$, consider $h_1 = h(x_1, y) = E_{x_1}[y] \oplus y$, let $e = E_{x_1}[y]$, then $h_1 = e \oplus y$

- then $h(c(x_1), c(y)) = E_{c(x_1)}[c(y)] \oplus c(y) = c(e) \oplus c(y) = e \oplus y = h_1$

- therefore, $h(c(x_1), x_2, \dots, x_k, c(y)) = h(x_1 x_2 \dots x_k, y)$

HW3: Problem 3b

- Goal: $h(x^1, y) = h(x^2, y) = h(x^3, y) = \dots$
 - find $h(x_1, y) = y \Leftrightarrow E_{x_1}[y] \oplus y = y \Leftrightarrow E_{x_1}[y] = 0 \Leftrightarrow y = D_{x_1}[0]$
 - $h(x_1 || x_1, y) = E_{x_1}[h(x_1, y)] \oplus h(x_1, y) = E_{x_1}[y] \oplus y = y$
 - thus $x_1, x_1 || x_1, x_1 || x_1 || x_1, \dots$ is the sequence we need

HW3: Problem 3c

- We don't control IV anymore
 - Observation, once we've found $h(x_1, y_0) = y = h(x_2, y)$
 - then $h(x_1 || x_2, y_0) = h(x_2, y) = y$, then adding any number of x_2 to the message, the hash is always y
 - we are given y_0 and we want $E_{x_1}[y_0] \oplus y_0 = y = E_{x_2}[y] \oplus y$
 - from $y = E_{x_2}[y] \oplus y$, $y = D_{x_2}[0]$, which is some random 64-bit string, when x_2 is random
 - given y_0 , $E_{x_1}[y_0] \oplus y_0$ is also random 32-bit string, when x_1 is random
 - by trying $O(2^{32})$ x_1 and x_2 , we have a reasonable probability that there is a collision

HW4: Problem 4

- This is known as the Merkle hash tree
- The proof is very similar to the one for the security of the Merkle-Damgard construction.

HW3: Problem 5

$$h_K(y_1, \dots, y_n) = e_K(y_1) + 3e_K(y_2) + \dots + (2n-1)e_K(y_n) \pmod{2^m}$$

- (a) (1,2) existential forger
 - request MAC $a = h_K(x, x, \dots, x)$, then $a = (n^2 e_K(x) \pmod{2^m})$
 - then $((n^2)^{-1} \pmod{2^m}) a \equiv e_K(x) \pmod{2^m}$
 - we can find $e_K(x)$
 - similarly, request MAC $b = h_K(y, y, \dots, y)$, we can find $e_K(y)$
 - then we can compute MAC for any message that is combination of x and y

HW3: Problem 5

- (b) $(1/8, 2)$ existential forger
 - Observation due to Adam: this is actually $(1/2, 2)$ forger
 - Let x' be $E[x] \bmod 2^m$ and y' be $E[y] \bmod 2^m$
 - $a = h_K(x, y) = E[x] + 3E[y] \bmod 2^m = x' + 3y' \bmod 2^m$
 - $b = h_K(x, y) = E[y] + 3E[x] \bmod 2^m = y' + 3x' \bmod 2^m$
 - $y' \equiv b - 3x'$, so $a \equiv x' + 3b - 9x'$, so $8x' \equiv 3b - a \pmod{2^m}$
 - Goal: output (x, x) , $4x' \bmod 2^m$
 - $2(4x' \bmod 2^m) \equiv 3b - a \pmod{2^m}$
 - there are two solutions for $4x' \bmod 2^m$

HW3: Problem 5

- (c) (1,3) selective forger
 - Given (y_1, y_2, \dots, y_n)
 - Ask MAC for $(x_1, y_2, \dots, y_n) = x_1' + (3y_2' \dots) \bmod 2^m = a$
MAC for $(x_1, x_2, \dots, x_n) = x_1' + (3x_2' \dots) \bmod 2^m = b$
MAC for $(y_1, x_2, \dots, x_n) = y_1' + (3x_2' \dots) \bmod 2^m = c$
 - MAC for $(y_1, y_2, \dots, y_n) = y_1' + (3y_2' \dots) \bmod 2^m$
 $= a + (y_1' - x_1') \bmod 2^m$
 $= a + (c - b) \bmod 2^m$