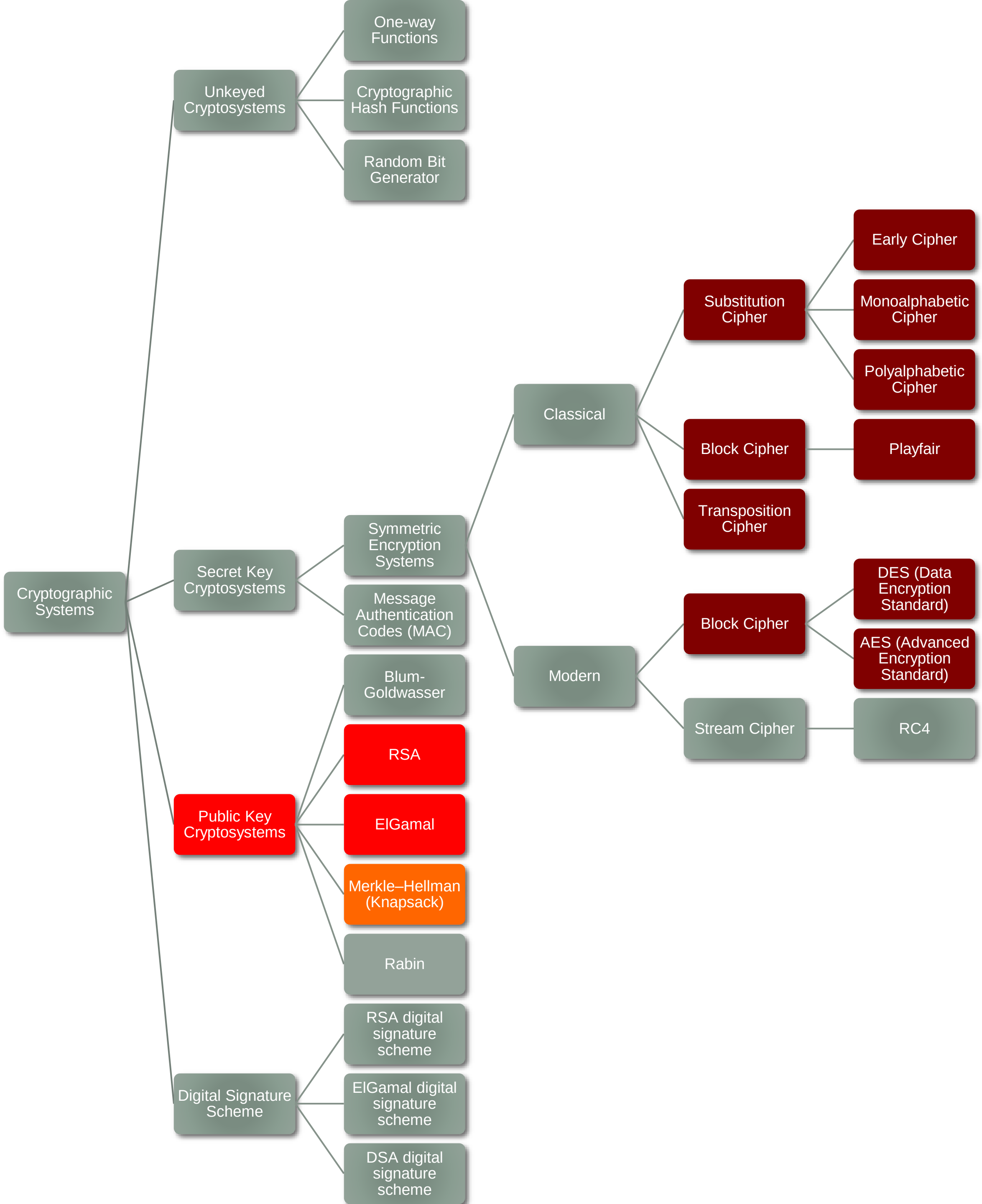
TUTORIAL 2

CSCI361 – Computer Security

Sionggo Japit

sjapit@uow.edu.au

12 February 2024


Cryptographic Systems
Unkeyed Cryptosystems
One-way Functions
Cryptographic Hash Functions
Random Bit Generator
Secret Key Cryptosystems
Symmetric Encryption Systems
Classical
Substitution Cipher
Early Cipher
Monoalphabetic Cipher
Polyalphabetic Cipher
Block Cipher
Playfair
Transposition Cipher
Modern
Block Cipher
DES (Data Encryption Standard)
AES (Advanced Encryption Standard)
Stream Cipher
RC4
Message Authentication Codes (MAC)
Public Key Cryptosystems
Blum-Goldwasser
RSA
ElGamal
Merkle–Hellman (Knapsack)
Rabin
Digital Signature Scheme
RSA digital signature scheme
ElGamal digital signature scheme
DSA digital signature scheme

TUTORIAL 2

Merkle-Hellman Knapsack

Merkle-Hellman knapsack

- **Merkle-Hellman Knapsack** Cryptosystem is a public-key cryptosystem described by Merkle and Hellman in 1978.
- This cryptosystem and various of its variants have been broken in the early 1980’s.
- It is based on the **subset sum** problem in combinatorics.
- The problem involves selecting a number of objects with given weights from a large set such that the sum of the weights is equal to a pre-specified weight.
- This is considered to be a difficult problem to solve in general. However, there are special cases that can be solved in polynomial time

Merkle-Hellman knapsack

- We define a list of sizes, $(s_1, \ldots, s_n)$ to be *superincreasing* if

$$s_j > \sum_{i=1}^{j-1} s_i$$

for $2 \leq j \leq n$.

- If the list of sizes is superincreasing, then the search version of the **Subset Sum** problem can be solved very easily in time $O(n)$, and a solution **x** (if it exists) must be unique.

Merkle-Hellman knapsack

- To deceive an attacker, the strategy therefore is to transform the list of sizes in such a way that it is no longer superincreasing.
- The attacker who does not know the transformation that was applied, is faced with what looks like a general, apparently difficult, instance of the subset sum problem.

Merkle-Hellman knapsack

- One suitable type of transformation is a modular transformation; that is, a prime modulus p is chosen such that

$$p > \sum_{i=1}^{n} s_i,$$

as well as a multiplier a, where $1 \le a \le p-1$.

Then we define $t_i = a s_i \mod p$, for $1 \le i \le n$.

- The list of sizes ***$t=(t_1,\ldots,t_n)$*** will be the public key used for encryption.
- The values a, p used to define the modular transformation are secret.

Merkle-Hellman knapsack

The encryption and decryption process:

1. Choose p and a, where $p > \sum_{i=1}^{n} x_i$, and a is random integer relatively prime to p; that is, $\gcd(p, a) = 1$. It is important that p and a are relatively prime, otherwise, inverse of a, which is needed to decrypt, cannot be obtained.
2. *Public key is computed as* $c_i = a \times x_i \; mod \; p$.
3. Secrete key is (x, p, a)
4. Sender compute $T = \sum(d_i \times c_i)$

Merkle-Hellman knapsack

5. Receiver decrypts using a^{-1} , found using extended Eucledean; that is, $y = a^{-1} \times T\ mod\ p$.
Receiver solves the instance $I = (b, y)$ of the subset sum problem and obtains the plaintext.

Merkle-Hellman knapsack

Example, suppose Alice wants to sent a plaintext ‘Hi’ to Bob using Bob’s public key b = (398, 144, 89, 775, 445, 73, 235, 195).

Merkle-Hellman knapsack

How Bob generates his public key?

1. Bob decides a super-increasing knapsack

   a = (2, 5, 9, 21, 45, 103, 215, 450).

2. Bob chooses a prime p = 851 such that p > (2 + 5 + 9 + 21 + 45 + 103 + 215 + 450 = 850, and a w = 199.

Bob checks if gcd(851,199) = 1 mod 851 using Extended Euclidean.

Merkle-Hellman knapsack

n1	n2	r	q	a1	b1	a2	b2
851	199	55	4	1	0	0	1
199	55	34	3	0	1	1	-4
55	34	21	1	1	-4	-3	13
34	21	13	1	-3	13	4	-17
21	13	8	1	4	-17	-7	30
13	8	5	1	-7	30	11	-47
8	5	3	1	11	-47	-18	77
5	3	2	1	-18	77	29	-124
3	2	1	1	29	-124	-47	201
2	1	0	2	-47	201	76	-325

GCD(851, 199) = 1 mod 851, thus *w* = 199 is relatively prime to 851.

w^{-1} = -325 mod 851 = -325 or 526 (Bob needs this to decrypt.)

Merkle-Hellman knapsack

3. Bob computes the public key $b_i = w \times a_i \bmod p$.

   b_1 = 199 x 2 mod 851 = 398
   b_2 = 199 x 5 mod 851 = 144
   b_3 = 199 x 9 mod 851 = 89
   b_4 = 199 x 21 mod 851 = 775
   b_5 = 199 x 45 mod 851 = 445
   b_6 = 199 x 103 mod 851 = 73
   b_7 = 199 x 215 mod 851 = 235
   b_8 = 199 x 450 mod 851 = 195

   The public key b = (398, 144, 89, 775, 445, 73, 235, 195) is obtained.

Merkle-Hellman knapsack

4. To encrypt, Alice converts the plaintext "Hi" into binary and computes the load T = å $(x_i$ ´ $b_i)$ using Bob's public key b; i.e., Alice encrypts the plaintext.

Plaintext (ASCII)	Decimal	binary
H	72	01001000
i	105	01101001

b = (398, 144, 89, 775, 445, 73, 235, 195)

$T_1 = (0 + 144 + 0 + 0 + 445 + 0 + 0 + 0) = 589$

$T_2 = (0 + 144 + 89 + 0 + 445 + 0 + 0 + 195) = 873$

Merkle-Hellman knapsack

5. Alice sends (589, 873) to Bob.
6. To decrypt, Bob computes $y = w^{-1}$ ´ $T \bmod p$.

$$y_1 = -325 \text{ ´ } 589 \bmod 851 = 50$$
$$y_2 = -325 \text{ ´ } 873 \bmod 851 = 509$$

Bob then recovers the plaintext by searching for instances (a, y).

Merkle-Hellman knapsack

a =	**2**	**5**	**9**	**21**	**45**	**103**	**215**	**450**	
y_1 = 50	0	1	0	0	1	0	0	0	= 50
y_2 = 509	0	1	1	0	1	0	0	1	= 509

Bob converts 01001000 and 01101001 back into ASCII, and the plaintexts 01001000 = H and 01101001 = i are recovered.