Cryptology part 2 Keränen /Teeriaho (Ramk 2007)

9 Authentication

■ 9.1 What is authentication ?

- **Definition**

Authentication is a process where one party becomes convinced of the identity of the other party through some evidence

- **Goals of authentication**

Result of an authentication is always either acceptance to the service or rejection from the service.

■ 9.2 Requirements for an authentication protocol

1) When A proofs his identity to B in the process, B cannot use the information he receives again in an authentication process to a third party C
(pin code, password must never be transferred in the protocol)
2) Probability of a situation, where a third party C could be accepted as A in the protocoll, is negligible.
3) The previous is true even if C had a possibility to "listen" lots of authentication processes between A and B.

Authentication is always a real time process in a sense that it requires that the user is active and uses the service at the moment.

Authentication can be based on

1) something you know: password, pin-code, private key
2) something you own: magnet card, smart card ,...
3) some unique property of the person: finger print, iris of the eye, voice ,...

Difference between authentication and digital signature

* Both protocols are related to each other, but authentication is more simple. In digital signature the changing element is the message. Digitally signed document should be juridically valid.
* In authentication the contents of the messages have no changing elements. Result is immediate: acceptance or denial of service.
* Authentications have no "life time", but digitally signed documents have.

■ 9.3 Weak authentication - password authentication

A fixed password is a traditional form of weak authentication. Used ID tells the identity and password gives a proof of the identity. Password is a shared secret between the used and the system. In a way it reminds of a symmetric key encryption.

Fixed password techniques

1) Uncrypted password files

User ID's and passwords are saved into a file which is not encrypted. However the file is read and write protected from all except administrators.
Weakness is that an administrator can read the passwords. Is he realiable?

- **2) Encrypted password files**

Passwords are encrypted with an *one way function, f.e hash -function* . Only the password's hash value is saved. It is not necessary to read protect the file.
When a user gives the password , only its hash value is transferred in an unprotected channel to the server.

- **3) Password "salting"**

D*ictionary attack* can be prevented by "salting" the password with a randomly chosen string before hashing. In fact also this salt must be saved somewhere, which makes also this method vulnerable.

- **Password attacks**

1. Reuse of passwords (not allowed in modern operation systems)
2. Brute Force attack
3. Dictionary attack

- **PIN – codes**

* Are used like passwords, but they are short (4 or 8 numbers)
* Pin - code is an additional safety measure f. e. in bank cards. Knowing the pin code is a proof of identity.
* Brute Force - attack is prevented in bank automats by allowing only 3 mistakes in giving the pin code.

9.4 One time passwords

Finnish internet banking authentication is based on one time passwords.
Types:

2.4.1 One time password lists

* The system and the user have both the same list of one time passwords. Each password can be used only in one authentication.
* Passwords must be used in the order they are in the list.
* Some banks use challenge -response lists. F.e Sampobanks list has pairs of numbers: 4 digit challenge number and 6 digit response number. The service presents the challenge number and the user has to answer with the corresponding response number to be accepted.
* No cryptographical methods are used in forming the list.
* A problem may be list management in the bank's servers.

2.4.2 One time password lists based on an one-way function

Lamport - list is an example.
* User and the system share a common key w.
* One way function H is used to calculate the sequence of passwords:
w , H(w) , H(H(w)) , ... H^t(w)
* The passwords must be used in the reverse order H^t(w), $H^{t-1}(w)$..

9.5 Challenge response authentication

Challenge response authentication means usually, that the service B send user A a challenge, which A encrypts and sends back to B.
B decrypts the response and compares the result with the challenge
The challenge is not really a message but most often
a random number (possibly some salt included).
Some times there is no challenge at all, only the response (example: time stamp)

Symmetric key challenge response authentication

(Kerberos, Needham Schroeder protocol)

In symmetric key systems both parties share a common key k.

Algorithm 1 *(One way authentication with random numbers):*
A = user (or smartcard B = service (automat)
E = encryption , k = symmetric key

1) B sends A a reandom number r.
2) A encrypts r with k and sends a response c =E(k,r,B*)
3) B decrypts D(k,c) = r' , B*
4) B compares: If r = r' , then A is authenticated.

B* is changing *salt*, which is optional.

Example. **GSM -algorithm A3.**

The SIM - card of the mobile phone and the operator possed the SIM key Ki.

1. Operator generates a random RAND and send it to the SIM - card
2. SIM - card uses algorithm A3 to encrypt RAND and send SRAND to the operator.
3. Operator also encrypts RAND using Ki.
4. If the results match , mobile phone is authenticated

Algorithm2 *(Time stamp authentication):*
A = user (or smartcard B = service (automat)
E = encryption , k = symmetric key
T = time stamp (date and time) from the users computer.

1) A encrypts T with the key k and sends B the cipher c =E(k,T,B*)
2) B decrypts the cipher D(k,c) =T , B*
3) If the decrypted time is within a predefined time window, user is authenticated.

B* is some changing element (salt) in the message to improve security.

Of both A , and B should authenticate each other, we speak about two way authentication.

Algorithm3 *(Two way authentication with random numbers):*

1) B sends A a random number rB.
2) A generates a random rA and sends B both numbers encrypted c = E(k, rA, rB, B*)
3) B decrypts the message and sends rA back E(k,rA,B*)
4) Both parties make comparisons and if they match, both are authenticated.

B* is salt.

- **Autentication using MAC**

MAC (usually HMAC) is an hash function , where symmetric key k is included in the message. MAC provides integrity check and authentication

Algorithm 4 *(MAC authentication):*

1) B sends A a random rB
2) A sends B a hash value MAC(k, rB)
3) B calculates also the hash value MAC(k,rB)
4) B compares. If hash values are same, A is authenticated.

Public key authentication

In public key systems authentication process is following:
Assume users A and B have public keys e_A and e_B , and secret keys d_A and d_B.

Algoritmi 5 *(Two way authentication with random numbers and RSA):*
Needham Schroeder Identification protocol

1) A sends B a random rA encrypted with B:s public key RSA(e_B, rA,A)
2) B decrypts and sends A two random numbers rA and rB as RSA(e_A, rA, rB)
3) A sends B number rB not encrypted.
4) If random numbers return unchanged, both are authenticated, because both were able to open the messages

Example: Let A:s RSA-keys be (nA, eA) = (91 , 31) and dA = 7.
B:s keys (nB,eB)=(187, 59) and dB = 19

```
nA = 91; eA = 31; dA = 7; nB = 187; eB = 59; dB = 19; n = Min[nA, nB];
rA = Random[Integer, n];
rB = Random[Integer, n];
v1 = PowerMod[rA, eB, nB];
a1 = PowerMod[v1, dB, nB];
v2 = {PowerMod[a1, eA, nA], PowerMod[rB, eA, nA]};
a2 = PowerMod[v2, dA, nA];
v3 = a2[[2]];
Print["A:s random challenge rA = ",
 rA, " is received by B as ", v1]
Print["B decrypts and gets ", a1]
Print["B:s random number ", rB,
 " and rA are received by A as: ", v2]
Print["A decrypts and gets (rA,rB)=: ", a2]
Print["A sends B number rB: ", a2[[2]]]
```

```
A:s random challenge rA = 61 is received by B as 156
```

```
B decrypts and gets 61
```

```
B:s random number 56 and rA are received by A as: {61, 56}
```

```
A decrypts and gets (rA,rB)=: {61, 56}
```

```
A sends B number rB: 56
```

It seems to work

9.6 Zero knowledge protocols

Fiat Shamir protocol and Schnorr ' s protocol

Zero knowledge protocols are not based on any public key cryptosystem or block ciphers. They remind of challnege response protocols, but they are planned so that no information about encryption keys is revealed.

Benefits of zero knowledge protocols:

1) Repeated use does not weaken the safety
2) No encryption algorithms are been used
3) Often very effective

Problems:

4) Based often to mathematical assumption like difficulty of finding square roots mod n.

Fiat Shamir protocol

User (smart card) has 2 public keys (n and ID), and one secret key stored in the chip.

Fiat Shamir - algorithm:

Secret key: s (Hidden in the card)
Public keys:

$n = p*q$ *product of two primes.*
$ID = s^2 \bmod n$

Authentication
1) Card generates a random r between 0 - (n-1) and sends its square $x = r^2$ mod n to the card reader.
2) Reader generates a random bit e (0 or 1) sending it to the card

3) Card calculates and send s $y = r\, s^e$ mod n to the reader

4) Card reader squares y^2 mod n and compares the result with ID^e mod n.
These should be same, because $y^2 = r^2 (s^2)^e = x\, ID^e$ mod n.

It can be shown , that probability that comparison gives TRUE , in case the card is false is 1/2. Repeating the protocol f.e five times and getting always TRUE, the probability of authenticity is already $1 - 2^{-5} = 97$ %.

Eacmple with Mathematica

- **1. Choose n = p*q**

```
n = 17 * 11

187
```

- **2. Choose secret key s ja and calculate its public square ID = s^2 mod n**

```
s = 89;
ID = Mod[s^2, n]

67
```

3. Challenge and comparison is repeated 10 times

```
lkm = 10;
While[lkm > 0,
 r = Random[Integer, {1, n - 1}];
 x = Mod[r^2, n];
 e = Random[Integer, 1];
 y = Mod[r * s^e, n];
 testi = Mod[y^2, n] == Mod[x * ID^e, n];
 lkm--;
 Print["r= ", r, " x= ", x, " e= ", e, " y= ", y, " test= ", testi];
]
```

```
r= 126 x= 168 e= 1 y= 181 test= True
```

```
r= 152 x= 103 e= 0 y= 152 test= True
```

```
r= 186 x= 1 e= 1 y= 98 test= True
```

```
r= 84 x= 137 e= 1 y= 183 test= True
```

```
r= 125 x= 104 e= 1 y= 92 test= True
```

```
r= 45 x= 155 e= 1 y= 78 test= True
```

```
r= 72 x= 135 e= 0 y= 72 test= True
```

```
r= 144 x= 166 e= 1 y= 100 test= True
```

```
r= 92 x= 49 e= 1 y= 147 test= True
```

```
r= 77 x= 132 e= 1 y= 121 test= True
```

Probability of genuinity of the card is $1 - 2^{-10}$ = 99,9 %

- **Schnorr protocol**

Bases on Discrete Logarithm Problem (DLP). A finite field F_q , where q is prime is needed. Let p be a prime factor of q-1 and g a primitive element of F_q .

Let a = $g^{\frac{q-1}{p}}$. Then integers 1, a, a^2, ... a^{p-1} are distinct.

Schnorr algorithm:

Given public:
Prime ***q*** *,*
Generator $\mathbf{g} \in Z_q$
A prime divisor p of q-1
Integer $a = g^{\frac{q-1}{p}} \bmod q$

Secret key x hidden in the smart card A
A's public key
$\boldsymbol{y = a^x} \textbf{ mod q}$
****
Authentication:
1) A generates random r between 0 ...(p-1) and sends B an integer $\rho = a^r \bmod q$
2) B sends A a random s between 0 ...(p-1)
3) A calculates u = r + s*x mod p and send it to B
4) B compares, if $a^u \bmod q = \rho * y^s \bmod q$
If comparison gives TRUE , A is authenticated

Reasoning:
$a^u = a^{r+s\,x} = a^r (a^x)^s = \rho\, y^s$
From traffic y , ρ , s and u in the channel one cannot calculate the secret x.
However the protocol proofs that A knows the secret key.

10. Example: Finnish electronic ID card

In many countries ID cards with cryptogtaphic properties have replaced traditional ID cards. These cards can be used for authentication in the web, electronic voting and in many other applications. Below is a description of the Finnish electronic ID card.

Properties:

Microchip: 16 kB memory, from which 5 kB for the program.

Contains RSA key pairs:

1 pair for digital signature and authentication
1 pair for encryption

Using of HST requires a card reader in the computer:

1. User contacts the service.
2. Service asks to place the card in the reader and give a PIN code
3. Authentication protocol which uses RSA starts