CS 758

Assignment 3

due by Noon on Monday, November 11, 2013

Questions

1. Suppose that the *Blom KPS* is implemented with security parameter k. Suppose that a coalition of k bad users, say $\mathrm{W}_1, \ldots, \mathrm{W}_k$, pool their secret information. Additionally, assume that a key $K_{\mathrm{U,V}}$ is exposed, where U and V are two other users.

   (a) Describe how the coalition can determine the polynomial $g_{\mathrm{U}}(x)$ by polynomial interpolation, using known values of $g_{\mathrm{U}}(x)$ at $k+1$ points.

   Answer: The coalition knows the k values

   $$\begin{aligned} g_{\mathrm{U}}(r_{\mathrm{W}_1}) &= g_{\mathrm{W}_1}(r_{\mathrm{U}}) \\ g_{\mathrm{U}}(r_{\mathrm{W}_2}) &= g_{\mathrm{W}_2}(r_{\mathrm{U}}) \\ \vdots &\quad \vdots \\ g_{\mathrm{U}}(r_{\mathrm{W}_k}) &= g_{\mathrm{W}_k}(r_{\mathrm{U}}). \end{aligned}$$

   Additionally, the coalition knows

   $$g_{\mathrm{U}}(r_{\mathrm{V}}) = K_{\mathrm{U,V}}.$$

   Therefore they have $k+1$ points on the polynomial $g_{\mathrm{U}}(x)$, which has degree at most k, and the coalition can compute $g_{\mathrm{U}}(x)$ by the interpolation formula.

   (b) Having computed $g_{\mathrm{U}}(x)$, describe how the coalition can compute the bivariate polynomial $f(x,y)$ by bivariate polynomial interpolation.

   Answer: The coalition knows the $k+1$ polynomials

   $$\begin{aligned} g_{\mathrm{W}_1}(x) &= f(x, r_{\mathrm{W}_1}) \\ g_{\mathrm{W}_2}(x) &= f(x, r_{\mathrm{W}_2}) \\ \vdots &\quad \vdots \\ g_{\mathrm{W}_k}(x) &= f(x, r_{\mathrm{W}_k}), \quad \text{and} \\ g_{\mathrm{U}}(x) &= f(x, r_{\mathrm{U}}). \end{aligned}$$

   Therefore the coalition can compute $f(x,y)$ by the bivariate interpolation formula.

(c) Illustrate the preceding two steps, by determining the polynomial $f(x, y)$ in the sample implementation of the *Blom KPS* where $k = 2$, $p = 1000003$, and $r_i = 11i$ $(1 \leq i \leq 4)$, supposing that

$$\begin{aligned} g_1(x) &= 292596 * x^2 + 502725 * x + 205896, \\ g_2(x) &= 379195 * x^2 + 870136 * x + 427928, \quad \text{and} \\ K_{3,4} &= 756408. \end{aligned}$$

Answer: First, $g_4(11) = g_1(44) = 789888$, $g_4(22) = g_2(44) = 833116$ and $g_4(33) = K_{3,4} = 756408$. Interpolating, $g_4(x) = 263968 * x^2 + 838469 * x + 626724$. Next, interpolating $g_1(x), g_2(x)$ and $g_4(x)$, we obtain

$$f(x, y) = 535417*x^2*y^2+430075*x*y*(x+y)+840971*x*y+213152*(x+y)+776524*(x^2+y^2)+902111.$$

(d) Compute $g_3(x)$ and $g_4(x)$.
Answer: We already computed $g_4(x)$. $g_3(x) = f(x, 33)$ is computed to be

$$36318 * x^2 + 315382 * x + 568204.$$

2. We describe a secret-key based three-party session key distribution scheme. In this scheme, K_{Alice} is a secret key shared by Alice and the TA, and K_{Bob} is a secret key shared by Bob and the TA.

Step 1. Alice chooses a random number, r_A. Alice sends ID(Alice), ID(Bob) and

$$y_A = e_{K_{\text{Alice}}}(\text{ID(Alice)} \parallel \text{ID(Bob)} \parallel r_A)$$

to Bob.

Step 2. Bob chooses a random number, r_B. Bob sends ID(Alice), ID(Bob), y_A and

$$y_B = e_{K_{\text{Bob}}}(\text{ID(Alice)} \parallel \text{ID(Bob)} \parallel r_B)$$

to the TA.

Step 3. The TA decrypts y_A using the key K_{Alice} and it decrypts y_B using the key K_{Bob}, thus obtaining r_A and r_B. It chooses a random session key, K, and computes

$$z_A = e_{K_{\text{Alice}}}(r_A \parallel K)$$

and

$$z_B = e_{K_{\text{Bob}}}(r_B \parallel K).$$

z_A is sent to Alice and z_B is sent to Bob.

Step 4. Alice decrypts z_A using the key K_{Alice}, obtaining K; and Bob decrypts z_B using the key K_{Bob}, obtaining K.

(a) State all consistency checks that should be performed by Alice, Bob and the TA during a session of the protocol.

Answer: The TA receives two identifying strings, say ID(Alice) and ID(Bob), in the clear. Then the TA should verify that $d_{K_{\text{Alice}}}(y_A)$ has the form

$$(\text{ID(Alice)} \parallel \text{ID(Bob)} \parallel r_A);$$

and $d_{K_{\text{Bob}}}(y_B)$ has the form

$$(\text{ID(Alice)} \parallel \text{ID(Bob)} \parallel r_B);$$

where r_A and r_B are arbitrary.

Alice should verify that $d_{K_{\text{Alice}}}(z_A)$ has the form $(r_A \parallel K)$, where r_A was the challenge that she chose in step 1.

Bob should verify that $d_{K_{\text{Bob}}}(z_A)$ has the form $(r_B \parallel K)$, where r_B was the challenge that he chose in step 2.

(b) The protocol is vulnerable to an attack if the TA does not perform the necessary consistency checks you described in part (a). Suppose that Oscar replaces ID(Bob) by ID(Oscar), and he also replaces y_B by

$$y_O = e_{K_{\text{Oscar}}}(\text{ID(Alice)} \parallel \text{ID(Bob)} \parallel r'_B)$$

in step 2, where r'_B is random. Describe the possible consequences of this attack if the TA does not carry out its consistency checks properly.

Answer: The TA receives identity strings ID(Alice), ID(Oscar), so it computes

$$d_{K_{\text{Alice}}}(y_A) = (\text{ID(Alice)} \parallel \text{ID(Bob)} \parallel r_A)$$

and

$$d_{K_{\text{Oscar}}}(y_O) = (\text{ID(Alice)} \parallel \text{ID(Bob)} \parallel r'_B).$$

The TA should reject because ID(Bob) occurs in the decryption of the strings y_A and y_O (rather than ID(Oscar)). If the TA fails to perform this check, it may go ahead and compute

$$z_A = e_{K_{\text{Alice}}}(r_A \parallel K)$$

and

$$z_O = e_{K_{\text{Oscar}}}(r'_B \parallel K).$$

z_A is sent to Alice and z_O is sent to Oscar. Alice and Oscar both obtain the key K, but Alice thinks she shares a session key with Bob. If Alice encrypts information using the key K, Oscar will be able to decrypt it and Bob will not be able to do so.

(c) In this protocol, encryption is being done to ensure both confidentiality and data integrity. Indicate which pieces of data require encryption for the purposes of confidentiality, and which ones only need to be authenticated. Rewrite the protocol, using MACs for authentication in the appropriate places.

Answer: In general, the first transmission of challenges and IDs does not need to be encrypted or authenticated. (Authentication does not achieve anything useful, because the adversary can reuse values from a previous session, even if they are authenticated.) Responses to challenges (including IDs) need to be authenticated (not encrypted). Session keys should be encrypted, and the encrypted session keys should be authenticated.

The protocol could be modified as follows:

1′. Alice chooses a random number, r_A. Alice sends ID(Alice), ID(Bob) and r_A to Bob.

2′. Bob chooses a random number, r_B. Bob sends ID(Alice), ID(Bob), r_A and r_B to the TA.

3′. The TA chooses a random session key, K, and computes

$$\begin{aligned} z_A &= e_{K_{\text{Alice}}}(K), \\ w_A &= MAC_{\text{Alice}}(\text{ID(Alice)} \parallel \text{ID(Bob)} \parallel z_A \parallel r_A), \\ z_B &= e_{K_{\text{Bob}}}(K), \quad \text{and} \\ w_B &= MAC_{\text{Bob}}(\text{ID(Alice)} \parallel \text{ID(Bob)} \parallel z_B \parallel r_B) \end{aligned}$$

(z_A, w_A) is sent to Alice and (z_B, w_B) is sent to Bob.

4′. Alice decrypts z_A using the key K_{Alice}, obtaining K; and Bob decrypts z_B using the key K_{Bob}, obtaining K. Alice and Bob also verify the MACs, w_A and w_B, respectively.

3. Discuss whether the property of perfect forward secrecy is achieved in *MTI/A0* for a session key that was established between U and V in the following two cases:

   (a) one LL-key a_{U} is revealed.

   Answer: Here we have perfect forward secrecy if the *Decision Diffie-Hellman* problem is infeasible in the group in question. An adversary can store the values s_{U} and s_{V} and they now know a_{U}. The session key is $K = (s_{\text{U}})^{a_{\text{V}}}(s_{\text{V}})^{a_{\text{U}}}$. The adversary can compute $(s_{\text{V}})^{a_{\text{U}}}$, so computing K is equivalent to computing $(s_{\text{U}})^{a_{\text{V}}} = \alpha^{r_{\text{U}}a_{\text{V}}}$. The adversary has $s_{\text{U}} = \alpha^{r_{\text{U}}}$ and $b_{\text{V}} = \alpha^{a_{\text{V}}}$, where a_{V} and r_{U} are not known. Computing any information about $\alpha^{r_{\text{U}}a_{\text{V}}}$ from this information cannot be done if the *DDH* is infeasible.

   (b) both LL-keys a_{U} and a_{V} are revealed.

   Answer: We do not have perfect forward secrecy in this situation, because a previous session key $K = (s_{\text{U}})^{a_{\text{V}}}(s_{\text{V}})^{a_{\text{U}}}$ can be computed by an adversary who has stored s_{U} and s_{V}.

4. (a) Suppose that the following are the nine shares in a $(5, 10)$-*Shamir Threshold Scheme* implemented in $\mathbb{Z}_{100000007}$:

i	x_i	y_i
1	1001	41884014
2	1002	90967621
3	1003	95742366
4	1004	29847132
5	1005	51923901
6	1006	35767422
7	1007	19576738
8	1008	7452181
9	1009	67898370
10	1010	84572698

Exactly one of these shares is defective (i.e., incorrect). Your task is to determine which share is defective, and then figure out its correct value, as well as the value of the secret. The "primitive operations" in your algorithm are polynomial interpolations and polynomial evaluations. Try to minimize the number of polynomial interpolations you perform.

Answer:

If we interpolate the first five shares, we get

$$f(x) = 53541799 * x^4 + 43007523 * x^3 + 84097111 * x^2 + 77652447 * x + 38495030.$$

If we then compute $f(6), \ldots, f(10)$, we found that $f(7), \ldots, f(10)$ yield the correct shares. $f(6) = 5617719$, so the sixth share is the incorrect share.

(b) Suppose that a (t, w)-*Shamir Threshold Scheme* has exactly one defective share, and suppose that $w - t \geq 2$. Describe how it is possible to determine which share is defective using at most $\lceil \frac{w}{w-t} \rceil$ polynomial interpolations. Why is this problem impossible to solve if $w - t = 1$?

Answer: Denote $\mu = \lceil \frac{w}{w-t} \rceil$. Let $T_1, \ldots, T_\mu$ be subsets of $\{1, \ldots, w\}$ such that

$$|T_j| = w - t$$

for $1 \leq j \leq \mu$, and

$$\bigcup_{j=1}^{\mu} T_j = \{1, \ldots, w\}.$$

This is not hard to do, for example by taking

$$T_1 = \{1, \ldots, w - t\}, T_2 = \{w - t + 1, \ldots, 2(w - t)\}, \ldots.$$

For each T_j, interpolate a polynomial $f_j(x)$ using the t shares in the set

$$S_j = \{s_i : i \notin T_j\}.$$

If S_j contains no bad shares, then $f_j(x)$ is the correct polynomial and $w - 1$ shares will lie on this polynomial. On the other hand, if S_j contains the bad share, then there are

at most $t-1$ good shares and one bad share (i.e., a total of t shares) that lie on the polynomial $f_j(x)$ (recall that two different polynomials of degree $t-1$ have at most $t-1$ points in common). Provided that $w-1 > t$, we can distinguish between these two cases. That is, an interpolated polynomial $f_j(x)$ is the correct one if and only if exactly one share does not lie on $f_j(x)$. Therefore we can determine the defective share after at most μ polynomial interpolations.

If $w = t+1$, then any interpolated polynomial will pass through exactly $w-1$ of the w shares. Here $t = w-1$ and the two cases considered above are indistinguishable. Therefore there is no way to tell which interpolated polynomial is correct.

(c) Suppose that a (t, w)-*Shamir Threshold Scheme* has exactly τ defective shares, and suppose that $w \geq (\tau+1)t$. Describe how it is possible to determine which shares are defective using at most $\tau+1$ polynomial interpolations.

Answer: For $i \leftarrow 1$ to $\tau+1$, perform the following steps:

step (1) Interpolate shares in the set $S_i = \{(i-1)t+1, \ldots, it\}$ to yield a polynomial $f_i(x)$.

step (2) Determine the set $B_i = \{j : f_i(x_j) \neq y_j\}$.

step (3) If $|B_i| \leq \tau$, then K is the constant term of f_i.

We now give a brief proof sketch that the algorithm described above will yield the correct value of the secret. Let $f(x)$ denote the polynomial used to generate the shares.

First, since $w \geq (\tau+1)t$, there exists at least one i such that S_i contains no bad shares. For any such i, we have that $f_i = f$, and at most τ of the w shares will not lie on f_i (i.e, $|B_i| \leq \tau$). In step (3), we will therefore obtain the correct value K as the secret.

On the other hand, suppose S_i contains at least one bad share. Note that S_i contains at most τ bad shares. Here, f_i agrees with at most $t-1$ good shares and at most τ bad shares, so $|B_i| \geq w-t-\tau+1$. We claim that $w-t-\tau+1 > \tau$. This inequality is equivalent to $w \geq t+2\tau$, which is true because $w \geq (\tau+1)t$ and $t \geq 2$. This proves the claim. It then follows that step (3) will fail for this value of i.