Cryptography, homework sheet 3

Due: 24 September 2015, 10:45

Team up in groups of two or three to hand in your homework. We do not have capacity to correct all homeworks individually. To submit your homework, email the programming part to `crypto15@tue.nl` and place the written part on the lecturer's table before the lecture. Do not email Tanja or put homework in mailboxes.
You may use computer algebra systems such as mathematica, gp, or sage or program in C, Java, or Python. Please submit your code as part of your homework. Make sure that your programs compile and run correctly; my students will not debug your programs. The program should be humanly readable.
If you are a student in the old masters system and for some reason cannot substitute 2MMC10 for the course (2WC09 or 2WC12) in your study program, please contact Tanja by email. To find out whether you can substitute the course, talk to your study advisor.

1. $3 \in \mathbb{F}_{1013}^*$ generates a group of order 1012, so it generates the whole multiplicative group of the finite field.

   Alice's public key is $h_a = 224$. Use ElGamal encryption to encrypt the messge $m = 42$ to her using the "random" value $k = 654$.

2. You find two signatures made by Alice. You know that she is using the ElGamal signature scheme over $\mathbb{F}_{2027}$ and that the order of the generator is $n = 1013$. The signatures are for $h(m_1) = 345$ and $h(m_2) = 567$ and are given by $(r_1, s_1) = (365, 448)$ and $(r_2, s_2) = (365, 969)$. Compute (a candidate for) Alice's long-term secret a based on these signatures, i.e. break the system.

3. $3 \in \mathbb{F}_{1013}^*$ generates a group of order $1012 = 4 \cdot 11 \cdot 23$. Solve the discrete logarithm problem $g = 3, h = 321$ by using the Pohlig-Hellman attack, i.e. find an integer $0 < k < 1012$ such that $h = g^k$ by computing first k modulo $2, 4, 11$, and 23 and then computing k using the Chinese Remainder Theorem.

Some people got stuck with the Chinese Remainder Theorem on the first sheet. Here is some text about it.

Theorem 1 (Chinese Remainder Theorem)
Let $r_1, \dots, r_k \in \mathbb{Z}$ and let $0 \neq n_1, \cdots, n_k \in \mathbb{N}$ such that the n_i are pairwise coprime. The system of equivalences

$$\begin{aligned} X &\equiv r_1 \bmod n_1, \\ X &\equiv r_2 \bmod n_2, \\ &\vdots \\ X &\equiv r_k \bmod n_k, \end{aligned}$$

has a solution X which is unique up to multiples of $N = n_1 \cdot n_2 \cdots n_k$. The set of all solutions is given by $\{X + aN | a \in \mathbb{Z}\} = X + N\mathbb{Z}$.

If the n_i are not all coprime the system might not have a solution at all. E.g. the system $X \equiv 1 \bmod 8$ and $X \equiv 2 \bmod 6$ does not have a solution since the first congruence implies that X is odd while the second one implies that X is even. If the system has a solution then it is unique only modulo $\mathrm{lcm}(n_1, n_2, \dots, n_k)$. E.g. the system $X \equiv 4 \bmod 8$ and

$X \equiv 2 \bmod 6$ has solutions and the solutions are unique modulo 24. Replace $X \equiv 2 \bmod 6$ by $X \equiv 2 \bmod 3$; the system still carries the same information but has coprime moduli and we obtain $X = 8a + 4 \equiv 2a + 1 \overset{!}{\equiv} 2 \bmod 3$, thus $a \equiv 2 \bmod 3$ and $X = 8(3b+2) + 4 = 24b + 20$. The smallest positive solution is thus 20.

We now present a constructive algorithm to find this solution, making heavy use of the extended Euclidean algorithm presented in the previous section. Since all n_i are coprime, we have $\gcd(n_i, N/n_i) = 1$ and we can compute u_i and v_i with

$$u_i n_i + v_i(N/n_i) = 1.$$

Let $e_i = v_i(N/n_i)$, then this equation becomes $u_i n_i + e_i = 1$ or $e_i \equiv 1 \bmod n_i$. Furthermore, since all $n_j | (N/n_i)$ for $j \neq i$ we also have $e_i = v_i(N/n_i) \equiv 0 \bmod n_j$ for $j \neq i$.
Using these values e_i a solution to the system of equivalences is given by

$$X = \sum_{i=1}^{k} r_i e_i,$$

since X satisfies $X \equiv r_i \bmod n_i$ for each $1 \leq i \leq k$.

Example 2 *Consider the system of integer equivalences*

$$\begin{aligned} X &\equiv 1 \bmod 3, \\ X &\equiv 2 \bmod 5, \\ X &\equiv 5 \bmod 7. \end{aligned}$$

The moduli are coprime and we have $N = 105$. For $n_1 = 3, N_1 = 35$ we get $v_1 = 2$ by just observing that $2 \cdot 35 = 70 \equiv 1 \bmod 3$. So $e_1 = 70$. Next we compute $N_2 = 21$ and see $v_2 = 1$ since $21 \equiv 1 \bmod 5$. This gives $e_2 = 21$. Finally, $N_3 = 15$ and $v_3 = 1$ so that $e_3 = 15$.
The result is $X = 70 + 2 \cdot 21 + 5 \cdot 15 = 187$ which indeed satisfies all 3 congruences. To obtain the smallest positive result we reduce 187 modulo N to obtain 82.

For easier reference we phrase this approach as an algorithm.

Algorithm 3 (Chinese remainder computation)
IN: *system of k equivalences as $(r_1, n_1), (r_2, n_2), \ldots (r_k, n_k)$ with pairwise coprime n_i*
OUT: *smallest positive solution to system*

```
1. $N \leftarrow \prod_{i=1}^{k} n_i$
2. $X \leftarrow 0$
3. for $i = 1$ to $k$
    (a) $M \leftarrow N \operatorname{div} n_i$
    (b) $v \leftarrow ((N_i)^{-1} \bmod n_i)$ (use XGCD)
    (c) $e \leftarrow vM$
    (d) $X \leftarrow X + r_i e$
```

4. $X \leftarrow X \bmod N$

The *Pohlig-Hellman attack* attack works in any group and is a way to reduce the reduce the hardness of the DLP to the hardness of the DLP in subgroups of prime order. In particular you'll see in the exercise that it works against the DLP in $\mathbb{F}_{1013}^*$ by solving DLPs in groups of size 2, 11, and 23. Here is the general description:
Let G be a cyclic group generated by g and let the challenge be to find $\log_g h = k$. Let the group order n factor as $n = \prod_{i=1}^r p_i^{e_i}$ where $p_i \neq p_j$ for $i \neq j$. Then k can be computed from the information

$$\begin{aligned} k &\equiv k_1 \bmod p_1^{e_1} \\ k &\equiv k_2 \bmod p_2^{e_2} \\ k &\equiv k_3 \bmod p_3^{e_3} \\ &\vdots \\ k &\equiv k_r \bmod p_r^{e_r} \end{aligned}$$

by using the Chinese remainder theorem. This is because the $p_i^{e_i}$ are coprime and their product is n. So, if one can find the DL modulo all $p_i^{e_i}$ one can compute the entire DL.
Put $n_i = n/p_i^{e_i}$. Since g has order n the element $g_i = g^{n_i}$ has order $p_i^{e_i}$. The element $h_i = h^{n_i}$ is in the subgroup generated by g_i and it holds that $h_i = g_i^{k_i}$, where $k_i \equiv k \bmod p_i^{e_i}$.
E.g. $\mathbb{F}_{16}^* = \langle g \rangle$ has 15 elements, so one can first solve the DLP $h = g^k$ modulo 3 and then modulo 5. For such small numbers one can simply compute h^5 and compare it to $1, g^5$, and g^{10} to find whether k is equivalent to $0, 1$, or 2 modulo 3. Then one compares h^3 to $1, g^3, g^6, g^9$, and g^{12} to see whether k is congruent to $0, 1, 2, 3$, or 4 modulo 5.
The same approach works also for $\mathbb{F}_{17}^*$ which has $16 = 2^4$ elements – but here one can do much better! Write $k = k_0 + k_1 2 + k_2 2^2 + k_3 2^3$. Then h^8 is either equal to 1 or to $-1 = g^8$ depending on whether k_0 is 0 or 1. Once that result is known we can compare $(h/g^{k_0})^4$ with 1 and -1 to find k_1 etc. So we can solve a much smaller DLP. Instead of going for k modulo $p_i^{e_i}$ at once we can first obtain k modulo p_i, then modulo p_i^2, then modulo p_i^3, etc. till $p_i^{e_i}$ by each time solving a DLP in a group of size p_i.
Numerical examples:

$\mathbb{F}_{11}^* = \langle 2 \rangle$, find k so that $3 = 2^k$. So $g = 2$ and $h = 3$. Compute $n_1 = 10/2 = 5$, $g^{n_1} = 2^5 = -1$, and $h^{n_1} = 3^5 = 1$ to see that $k \equiv 0 \bmod 2$. Then compute $n_2 = 10/5 = 2$, $g^{n_2} = 2^2 = 4, g^{2n_2} = 2^4 = 5, g^{3n_2} = 2^6 = 9$, and $g^{4n_2} = 2^8 = 3$ and compare that to $h^{n_2} = 3^2 = 9$ to see that $k \equiv 3 \bmod 5$. These two congruences imply that $k = 8$ and indeed $g^8 = h$.
$\mathbb{F}_{17}^* = \langle 3 \rangle$, find k so that $7 = 3^k$. So $g = 3$ and $h = 7$. In this example we will obtain k one bit at a time. First compare $h^8 = 7^8 = -1$ to 1 and -1 to see that $k \equiv 1 \bmod 2$. Then compute $h/g = 8$ and then $(h/g)^4 = -1$, so also the next bit is 1 and we see $k \equiv 3 \bmod 4$. Then compute $h/g^3 = 16$ and then $(h/g^3)^2 = 1$ to see that the next bit is 0, so $k \equiv 3 \bmod 8$. Finally, since $h/g^3 = 16 = -1$ we see that the highest bit is 1, so $k \equiv 11 \bmod 16$ and indeed $3^{11} = 7$. This solved the DLP in $\mathbb{F}_{17}^*$ with just 4 very easy computations and comparisons. So computing DLs in fields $\mathbb{F}_p$ with $p = 2^r + 1$ is easy.