Introduction à la cryptologie

TD n° 8 : Attacks in the RSA and DLOG setting.

Exercice 1 (Low Hamming Weight DLOG). Let $G$ be some group of order $q$ with fixed generator $g$. Given $[m] = \{1, \ldots, m\}$ and $t \in \mathbb{N}$, we denote by $\binom{[m]}{t}$ all subsets of $[m]$ of size $t$. Next, given some $m, t \in \mathbb{N}$, we define $\text{val} : \binom{[m]}{t} \to \mathbb{N}$ as $\text{val}(Y) = \sum_{i \in Y} 2^{i-1}$. Let us consider algorithm 1.

1. Replace TODO in line 10 such that the algorithm returns the discrete logarithm $x$ of $h$.
2. Show alg. 1 terminates if $x = \text{dlog}_g(h)$ is indeed of bit-length at most $m$ and hamming weight $t$.
3. Show that the algorithm runs in $\Theta\left(\frac{m}{t/2}\right)$ group exponentiations.
4. Show that the algorithm requires storage of $\Theta\left(\frac{m}{t/2}\right)$ group elements.

Algorithm 1 Low Hamming DLOG

Require: $m, t, g, h \in G$, where we assume that $x = \text{dlog}_g(h)$ has bit-length at most $m$ and hamming weight $t$ for some even $t$.

1: Initialize hash table $H$
2: for all $Y_1 \in \binom{[m]}{t/2}$ do
3: $y_1 \leftarrow g^{\text{val}(Y_1)}$
4: $H$.put($y_1, Y_1$).
5: end for
6: for all $Y_2 \in \binom{[m]}{t/2}$ do
7: $y_2 \leftarrow h \cdot (g^{-1})^{\text{val}(Y_2)}$
8: if $H$.contains_key($y_2$) then
9: $Y_1 \leftarrow H$.get($y_2$)
10: return TODO
11: end if
12: end for

Exercice 2 (Textbook RSA). We analyze textbook RSA (the variant given in the first course).

1. Recall the encryption and signature schemes.
2. Show that RSA encryption is insecure. That is, given a ciphertext $c$ on either $m_0$ or $m_1$, an attacker knowing $m_0$ and $m_1$ can decide whether it was given $c_0$ or $c_1$.
3. Show that any deterministic encryption scheme is insecure.
4. Show that RSA signatures are insecure. That is, given access to a signature oracle, compute a signature for an unqueried message. Can you forge a signature without oracle?
5. Propose mitigations against these attacks.
6. Show that RSA signatures are secure if message is first hashed. In the proof, you can model the hash function as a programmable random oracle (see last TD).

Exercice 3 (Bleichenbacher attack on RSA PKCS). We investigate security of RSA encryption in the setting where encrypted messages have a fixed format and an oracle is available that returns whether the encrypted message has the correct format. The message format for RSA PKCS is given below

\[
m = 00\|02\|\text{pad}\|00\|\text{data},
\]

where data is the message to be encrypted and pad is a padding. Let $n, e$ be an RSA public key with byte length $k$ of $n$, so $2^{k(1-k)} \leq n < 2^{k^2}$. A data block data of |data| $\leq k-11$ bytes is padded with a random pad of $k-3-\text{data}$ non-zero bytes. Then, 00\|02\|\text{pad}\|00\|\text{data} is encrypted via RSA as usual. A ciphertext is said to be PKCS conforming if the decrypted value is of the above message format. Set $B = 2^{8(k-2)}$.

Let us assume we have access to an oracle $O(\cdot)$ that given a ciphertext $c$ outputs whether the corresponding plaintext is PKCS conforming.
1. What could be the purpose of this padding?
2. Is the assumption of having such an oracle realistic in practice?
3. Show that for some PKCS conforming $m$, we have $m \in \{2B, 3B - 1\}$.

Next, we consider algorithm 2.

**Algorithm 2** Bleichenbacher

**Require:** Ciphertext $c_0 = m_0^c \mod n$ with PKCS conforming $m_0$.

1. Set $M_0 = \{\lfloor 2B, 3B - 1 \rfloor\}, s_0 = 1$ and $i = 1$.
2. **if** $i = 1$ **then**
3. search for smallest $s_1 \geq n/(3B)$ such that $c_0(s_1)^c \mod n$ is PKCS conforming.
4. **end if**
5. **if** $i > 1$ and $|M_{i-1}| \geq 2$ **then**
6. Find smallest $s_i > s_{i-1}$ such that $c_0(s_i)^c \mod n$ is PKCS conforming.
7. **else**
8. Parse $M_{i-1} = \{[a, b]\}$.
9. Choose small integer values $(r_i, s_i)$ such that

\[
   r_i \geq 2 \cdot \frac{b \cdot s_{i-1} - 2B}{n} \quad \text{and} \quad \frac{2B + r_i \cdot n}{b} \leq s_i < \frac{3B + r_i \cdot n}{a},
\]

until $c_0(s_i)^c \mod n$ is PKCS conforming.
10. **end if**
11. Set

\[
   M_i = \bigcup_{(a, b, r)} \left\{ \left[ \max \left( a, \left\lceil \frac{2B + rn}{s_i} \right\rceil \right), \min \left( b, \left\lfloor \frac{3B - 1 + rn}{s_i} \right\rfloor \right) \right] \right\}
\]

for all $[a, b] \in M_{i-1}$ and $\frac{a \cdot s_i - 3B + 1}{n} \leq r \leq \frac{b \cdot s_i - 2B}{n}$.
12. **if** $|M_i| = \{[a, b]\}$ **then**
13. **return** $m = a(s_0)^{-1}$
14. **else**
15. Set $i = i + 1$ and go to line 2
16. **end if**

4. We want that $s_1 \in \{2, n\}$ is the smallest value such that $c_0(s_1)^c$ is PKCS conforming. Explain why we nevertheless start at $n/(3B)$ in line 3.
5. We are interested in line 11. Show that if $m_0 \in M_{i-1}$, then there is some $r$ in the desired range which guarantees that $m_0 \in M_i$ (cf. equation 2). **Hint**: use that $m_0s_i$ is PKCS conforming to find an appr. $r$.
6. Conclude that algorithm 2 outputs $m_0$ if it terminates.
7. Adapt the attack to forge an RSA signature on an arbitrary message $c \in \mathbb{Z}_n$ with the help of a padding oracle (if the same secret key is used for encryption and decryption).
8. Estimate rough upper and lower bounds for $\Pr[A]$, where $A$ is the event that some random $m$ is PKCS conforming for $k \geq 64$.

**Hint**: You can use that $-19 < \log_2(\Pr[A]) < -8$ in the following.
9. Show that there are at most $\left\lceil \frac{\omega_i B}{n} \right\rceil$ intervals in $M_i$.

We denote by $\omega_i$ the number of intervals in iteration $i$. We assume that $\omega_1 \leq 1 + 2^{i-1}s_i(\frac{B}{n})^i$. We further assume that all $s_i$ chosen in the algorithm have a probability of $\Pr[A]$ to be PKCS conforming, even though the $s_i$ are not independently uniform.
11. These are a lot of heuristic assumptions... Why are assumptions fine in this context (but less so in the context of security proofs) ?
12. Argue that the condition in line 5 is unlikely to be true more than once.
13. Argue that it is quite likely to find a suitable pair $(r_i, s_i)$ satisfying condition (1) in line 9. Note that $r_i$ is chosen first. **Hint**: Show that for some $r_i$, the corresponding interval of $s_i$ has length around $1/3$.
14. Argue that the algorithm terminates.
15. **Bonus**: estimate the number of required oracle queries.
16. **Bonus**: argue that indeed $\omega_i \leq 1 + 2^{i-1}s_i(\frac{B}{n})^i$, assuming that the intervals $I_r \in M_i$ are uniformly distributed.