Lecture Notes: 01/31/2002

1 Modified Goldreich-Levin Theorem

Given a trapdoor permutation and a hardcore predicate, we have semantically secure public key encryptions. For RSA, if it's secure, then given the encoding of $x^e \bmod N$, it is hard to guess $x \bmod 2$.

General Result: Every trapdoor permutation has a hard-core predicate. We therefore need not go on a case-by-case basis. This is due to a slight modification of the Goldreich-Levin theorem:

THEOREM 1
Suppose (G, F, I) is a family of trapdoor permutations. Consider (G, F', I') where $F'((x, r), k) = F(x, k), r$ and $I'((z, r), t_k) = I(z, t_k), r$. Then

$$B(x, r) = \sum_i x_i \cdot r_i \bmod 2$$

is a hard-core predicate for (G, F', I').

We allow F' and I' to have 2 (sets of) input parameters. However, we can still apply the original inverting algorithm.

For random x, r, for $(k, t_k) \in G$, given $F(x, k), r$, it is hard to guess $\sum_i x_i \cdot r_i \bmod 2 = x \cdot r$. So if we can break this, a family of trapdoor permutations can be broken, also.

2 Proof

By contradiction, we assume this is not hard-core predicate, and we would like to show that F can be broken easily.

2.1 Probability of breaking $> \frac{1}{2} + \varepsilon$

Suppose there is an adversary A such that

$$\Pr_{\substack{x,r \\ (k,t_k)\in G}}[A(F(x,k),r,k) = x \cdot r] > \frac{1}{2} + \varepsilon \quad (*)$$

.

We would like to show that

$$\Pr_{(k,t_k)\in G}[\Pr_{x,r}[A(F(x,k),r,k) = x \cdot r] > \frac{1}{2} + \frac{\varepsilon}{2}] > \frac{\varepsilon}{2}$$

Proof: Suppose, to the contrary, that

$$\Pr_{(k,t_k)\in G}[\Pr_{x,r}[A(F(x,k),r,k) = x \cdot r] > \frac{1}{2} + \frac{\varepsilon}{2}] < \frac{\varepsilon}{2}$$

We can then rewrite (*):

$$\begin{aligned}
&\Pr_{\substack{x,r \\ (k,t_k)\in G}}[A(F(x,k),r,k) = x \cdot r] > \frac{1}{2} + \varepsilon = \\
&\quad = \sum_{(k,t_k)\in G} \Pr[G\text{outputs}(k,t_k)] \cdot \Pr_{x,r}[A(F(x,k),r,k) = x \cdot r] \\
&\quad = \underbrace{\sum_{\substack{(k,t_k)\in G \\ \ni \Pr_{x,r}[A(F(x,k),r,k)=x\cdot r] > \frac{1}{2}+\varepsilon}} \Pr[G \text{ outputs } (k,t_k)] \cdot \Pr_{x,r}[A(F,x,k),r,k) = x \cdot r +}_{<\frac{\varepsilon}{2}} \\
&\quad + \underbrace{\sum_{\substack{(k,t_k)\in G \\ \ni \Pr_{x,r}[A(F(x,k),r,k)=x\cdot r] \le \frac{1}{2}+\varepsilon}} \Pr[G \text{ outputs } (k,t_k)] \cdot \Pr_{x,r}[A(F,x,k),r,k) = x \cdot r}_{\le \frac{1}{2}+\varepsilon} \\
&\quad < \frac{1}{2} + \varepsilon
\end{aligned}$$

which contradicts (*). □

- With probability $\frac{\epsilon}{2}$, k is good (i.e., $A(F(x,k),r,k) = xr$). When k is good,

  $$\Pr_x[\Pr_r(A(x,k),r,k) = x \cdot r] \geq \frac{1}{2} + \frac{\epsilon}{4}] \geq \frac{\epsilon}{4}$$

- With probability $\leq \frac{\varepsilon^2}{8}$, x and k are good (i.e., $A(F(x,k),r,k) = xr$). Define $A'(r) = A(F'(r,k),r,k)$. When x and k are good,

  $$\Pr_r[A'(r) = x \cdot r] \geq \frac{1}{2} + \frac{\varepsilon}{4}$$

  So using A', we want to compute x (this is correct $\geq \frac{1}{2}$ of the time) in time $\text{poly}(n, \frac{1}{\varepsilon}), n = \mid x \mid$.

  $\Rightarrow$ there is a reasonable probability we can break $F \Rightarrow$ not a family of trapdoor permutations

2.2 Probability of breaking = 1

- Suppose A' always works: $\Pr_r[A'(r) = x \cdot r] = 1$. Then we can use $e_i = (0, \ldots, 0, \overbrace{1}^{i^{th}\text{ component}}, 0, \ldots, 0)$ to find the n individual components, x_i, of x (thus completely reconstructing x):

$$\begin{array}{rcccl} A'(e_1) & = & x \cdot e_1 & = & x_1 \\ A'(e_2) & = & x \cdot e_2 & = & x_2 \\ & & \cdots & & \\ A'(e_n) & = & x \cdot e_n & = & x_n \end{array}$$

- If $\Pr_r[A'(r) = x \cdot r] \geq 1 - \frac{1}{10n}$, can still manage fairly well: Pick n random vectors r_i:

$$\begin{array}{rcl} A'(r_1) & = & x \cdot r_1 \\ A'(r_2) & = & x \cdot r_2 \\ & & \cdots \\ A'(r_n) & = & x \cdot r_n \end{array}$$

  Each component is likely to be correct ($> 90\%$), and all are linearly independent $\Rightarrow$ can solve the system of equations to find x.

2.3 Probability of breaking $\geq \frac{3}{4} + \varepsilon$

Suppose the adversary's algorithm only works $\Pr_r[A'(r) = x \cdot r] \geq \frac{3}{4} + \varepsilon$. Now for all r,

$$x_i = x \cdot e_i = x \cdot r + x \cdot (e_i - r) = x \cdot (r + e_i - r)$$

If we pick a random r and compute $A'(r) + A'(e_i - r)$,

$$\Pr[\underbrace{A'(r)}_{\text{correct with } \frac{3}{4}+\varepsilon} + \underbrace{A'(e_i - r)}_{\text{correct with } \frac{3}{4}+\varepsilon} = x_i] \geq \frac{1}{2} + 2\varepsilon$$

In other words, the adversary makes an error less than $\frac{1}{2} - 2\varepsilon$ amount of the time.
So to find x:
  Repeat k times

- pick random r and compute $A'(r) + A'(e_i - r)$
- $\Pr[A'(r) + A'(e_i - r) = x_i] \geq \frac{1}{2} + 2\varepsilon$
- output the result that comes out more often

Comments

- Probability of error $\leq 2^{-\Omega(\epsilon^2 \cdot k)}$.

- If we set $k = \Theta(\frac{\log n}{\varepsilon^2}), \Pr[error] \leq \frac{1}{n^2}$ If repeat for $i = 1, \ldots, n$, all steps will be correct with probability $\geq 1 - \frac{1}{n}$
- running time: $\Theta(\frac{n \log n}{\varepsilon^2})$ invocations of A'

2.4 Back to probability of breaking $\geq \frac{1}{2} + \varepsilon$

Now consider the case where the probability the adversary can break the code is $\Pr_r(A'(r) = x \cdot r) \geq \frac{1}{2} + \epsilon$. This is the strongest case in our analysis.

2.4.1 $k = \Theta(\frac{\log n}{\varepsilon^2})$

- Pick k vectors at random: $r_1, r_2, \ldots, r_k$
- "Guess" value of $x \cdot r_1, \ldots, x \cdot r_k$ (only one will be correct, need to repeat 2^k times)
- Estimate x_i as majority value of

$$\begin{array}{lll}
x \cdot r_1 + A'(e_i - r_1) & \text{probability } \frac{1}{2} + \varepsilon & \text{of getting } x_i \\
x \cdot r_2 + A'(e_i - r_2) & \text{probability } \frac{1}{2} + \varepsilon & \text{of getting } x_i \\
\ldots & & \\
x \cdot r_k + A'(e_i - r_k) & \text{probability } \frac{1}{2} + \varepsilon & \text{of getting } x_i
\end{array}$$

$$\begin{array}{lll}
\text{Overall probability of error is} & \leq 2^{-\Omega(\varepsilon^2 k)} & \\
& \leq \frac{1}{n^2} & k = \Theta(\frac{\log n}{\varepsilon^2})
\end{array}$$

- Repeat for all i
- We repeat all this $n^{O(\frac{1}{\varepsilon^2})}$ times.
- Try all possible guesses, find different x

We have fallen slightly short of completely decoding the message, as we have not found x itself, but only candidate x's. However, since F is easy to evaluate, we need only evaluate F for all candidate x's, and pick the right one.

The last optimization we would like to do is to get the running time to $\text{poly}(n, \frac{1}{\varepsilon})$.

2.4.2 $k \approx \Theta(\log \frac{n}{\varepsilon^2})$

- Pick k vectors at random: $r_1, \ldots, r_k$
- Guess $r_1 x, \ldots, r_k x$

- For all $S \in \{1, \ldots, k\}, S \neq \emptyset$

$$r_S = \sum_{i \in S} r_i$$

can compute $x \cdot r_S = \sum_{i \in S} x \cdot r_i$

- For every $i \in \{1, \ldots, n\}$ compute x_i as the majority of $\underbrace{r_S}_{2^{k-1}} \cdot x + A'(e_i - r_S)$

over all $S \neq \emptyset$

Since $r_1, \ldots, r_k$ are random, $e_i - r_S$ is random. For every $S \neq T$, r_S and r_T are uniformly distributed and random.

$$\begin{aligned} r_S &= r_{S-T} + r_{S \cap T} \\ r_T &= r_{T-S} + r_{S \cap T} \end{aligned}$$

There are 2^{k-1} events that are pairwise independent; the computation is correct, and the probability is $\geq \frac{1}{2} + \varepsilon$. The probability of a majority of

incorrect computations is $\leq O(\frac{1}{\varepsilon^2 2^{k-1}}) < \frac{1}{n^2}$ if $k = \log(\frac{n^2}{\varepsilon^2}) + 1$
variation $\sim 2^k$
deviation $\sim \varepsilon 2^k$ $\quad \frac{\text{var}}{\text{dev}^2} \sim \frac{1}{\varepsilon^2 2^k}$

The following actions must be performed 2^k times:

- Pick k random vectors
- Invoke A' for all j $\Rightarrow$ $\frac{n^3}{\varepsilon^4}$ invocations of A', constant probability of finding x:

  probability of finding $x = 1 - \delta$ $\Rightarrow \frac{n^3}{\delta^2 \varepsilon^4}$ invocations
- Evaluate F on each x

If we can break the hard-core predicate for the inner product, we can invert F in poly$(n, \frac{1}{\varepsilon})$

$\Rightarrow$ sequence of steps from trapdoor permutation to a completely secure system $\Rightarrow$ not a family of trapdoor permutations, contradition.

2.4.3 $k = \log(\Theta(\frac{1}{\varepsilon^2}))$

We now try to reduce the number of candidate x values. If $k = \log(\Theta \frac{1}{\varepsilon^2})$, and have $r_1, \ldots, r_k$, guess $x \cdot r_1, \ldots, x \cdot r_k$

r_S for $S \in \{1, \ldots, k\}$

compute $x \cdot r_S = \sum_{i \in S} x \cdot r_i$ and for all r, $r \cdot x$

$$\begin{array}{ll} \text{majority} & r_1 \cdot x + A'(r - r_1) \\ & r_2 \cdot x + A'(r - r_2) \\ & \ldots \\ & r_k \cdot x + A'(r - r_k) \end{array}$$

This is correct with $\geq \frac{7}{8}$ over the choices of r (and $r_1, \ldots, r_k$))

$\Rightarrow$ final list of x's is smaller, but the running time is roughly the same

3 Length of encoding

3.1 Encoding scheme 1

Given $F(\cdot, k)$, $B(\cdot, k)$, and $m \in \{0,1\}^n$,
pick $x_1, \ldots, x_n$
send

$$(F(x_1, k), m[1] \oplus B(x_1, k)), \ldots, (F(x_n, k), m[n] \oplus B(x_n, k))$$

REMARK 1 If we wanted to use the Goldreich-Levin theorem, we would also choose $r_1, \ldots, r_n$. We then send

$$(F(x_1, k), r_1, m[1] \oplus x_i r_i), \ldots, (F(x_n, k), r_n, m[n] \oplus x_n r_n)$$

If the encoding is 1000 bits, the encoded message will be 1000 times longer than the original message. We must perform the decoding algorithm (e.g. RSA) 1000 times.

3.2 Encoding scheme 2

Given $F(\cdot, k) = f()$, $B(\cdot, k) = B()$,$m \in \{0,1\}^n$,
pick $x_1, \ldots, x_n$
send

$$B(x) \oplus m[1], B(f(x)) \oplus m[2], \ldots, B(f^{(n-1)}(x)) \oplus m[n], f^{(n)}(x)$$

where $f^{(n)}(x) = f\underbrace{(f(f(\ldots(x)\ldots)))}_{n \text{ times}}$

This reduces the message encoding length, but still have to evaluate the decoding function the same number of times. With the trapdoor, it is easy to get x, and thus m. Without the trapdoor, this is message indistinguishable.

For proof:
For every n, for every adversary A, for all m, m'

$$\begin{aligned} |\Pr[A(b_1, \ldots, b_{2n}) = 1] - \Pr[A(E(m)) = 1]| &\leq \varepsilon \\ |\Pr[A(b_1, \ldots, b_{2n}) = 1] - \Pr[A(E(m')] = 1| &\leq \varepsilon \\ |\Pr[A(E(m)) = 1] - \Pr[A(E(m')) = 1]| &\leq \varepsilon \end{aligned}$$

4 Attacks

4.1 Lunch-time attack

For a period of time, an adversary can get the decryption of chosen messages. (If the system can be broken like this, it is bad.) For a simple message, message indistinguishability is "enough" security. But when more periods are being sent, this type of attack is a problem.
Consider the case where we are using Goldwasser-Micali encryption with Rabin's function.

$$\begin{aligned} \text{inverting Rabin's function} \quad &\equiv \text{factoring} \\ &\equiv \text{breaking hard-core predicate } x \bmod 2 \text{ (parity)} \\ &\equiv \text{decoding messages in Goldwasser-Micali based on Rabin's function} \end{aligned}$$

For a message of l bits, encrypted message has l bits + overhead in the size of the range of $f(x)$.

4.2 Adaptive chosen ciphertext attack

Informally, the adaptive chosen ciphertext attack is as follows: Eve gets $E(m, p_k), p_k$ and can then ask for a decoding $D(c', s_k)$ where $c \neq E(m, p_k)$. If the system is secure, she gets no information about m. I.e., the ability to decrypt other ciphertext gives no information about m. (Goldwasser-Micali does not satisfy this notion of security.)