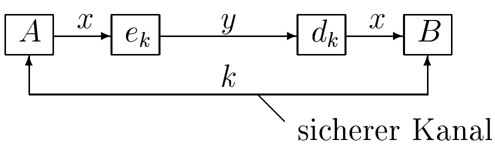
9 Kryptographische Grenzen der Erlernbarkeit

9.1 Klassische Kryptosysteme

In einem klassischen (symmetrischen) Kryptosystem will Alice über einen unsicheren Kanal eine vetrauliche Nachricht x an Bob senden. Die Nachricht x wird auch *Klartext* genannt. Hierzu bedienen sich Alice und Bob einer *Verschlüsselungsfunktion* e_k, die den Klartext x in einen *Kryptotext* $y = e_k(x)$ überführt, und einer korrespondierenden *Entschlüsselungsfunktion* d_k. Sowohl e_k als auch d_k hängen von einem Schlüssel k ab, der nur Alice und Bob bekannt sein darf. Daher muss k vor der Nachrichtenübermittlung über einen *sicheren Kanal*, wie zum Beispiel durch ein geheimes Treffen, ausgetauscht werden. Daraus ergeben sich die folgenden Anforderungen an ein Kryptosystem.

1. $d_k(e_k(x)) = x$ für alle Klartexte x und Schlüssel k
2. e_k und d_k sind effizient berechenbar
3. Ohne Kenntnis des Schlüssels k lassen sich aus dem Kryptotext $y = e_k(x)$ keine Rückschlüsse auf den Klartext x ziehen.



Beispiel 9.1 (One-time Pad). Beim One-time Pad wird zur Übertragung eines Klartexts $x \in \{0,1\}^n$ ein zufälliger Schlüssel $k \in_U \{0,1\}^n$ gewählt. Die Ver- und Entschlüsselungsfunktionen sind

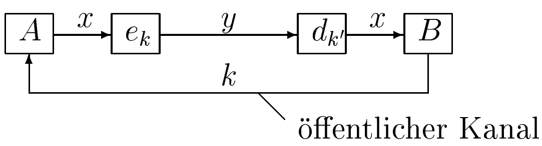
$$e_k(x) = x \oplus k \quad \text{und} \quad d_k(y) = y \oplus k,$$

wobei "$\oplus$" das bitweise XOR zweier Binärwörter gleicher Länge bezeichnet. Es ist klar, daß $d_k(e_k(x)) = x$ für alle Klartexte $x \in \{0,1\}^n$ und $k \in \{0,1\}^n$

gilt, und daß e_k und d_k effizient berechenbar sind. Somit sind die ersten beiden Anforderungen an ein Kryptosystem erfüllt. Für die dritte Anforderung beachten wir, daß für einen zufälligen, unter Gleichverteilung gewählten Schlüssel k auch der Kryptotext $y = e_k(x)$ eine zufällige, unter Gleichverteilung gewählte Bitfolge ist. Ein Beobachter "sieht" also nur eine zufällige Münzwurffolge und kann daher auch mit unbeschränkten Ressourcen aus y keine Information über den Klartext x gewinnen.

9.2 Das RSA-Kryptosystem

Nachteil beim One-time Pad, kurze Beschreibung von Public-Key Verfahren ...



Beim RSA-Kryptosystem werden die beiden Schlüssel k und k' von B in der folgenden Weise bestimmt.

- Wähle (genügend große) Primzahlen p und q, und setze $N = pq$
- Wähle b mit $1 < b < \varphi(N)$ und $ggT(b, \varphi(N)) = 1$, wobei $\varphi(N) = (p-1)(q-1)$
- Berechne $a < \varphi(N)$ mit $ab \equiv 1 \pmod{\varphi(N)}$

Die Existenz von a folgt aus $ggT(b, \varphi(N)) = 1$ und kann mit Hilfe des "Extended Euklid"-Algorithmus effizient berechnet werden. Der öffentliche Schlüssel ist dann $k = (N, b)$, und die Faktoren p, q und a bilden den geheimen Schlüssel k'. Die Ver- und Entschlüsselungsfunktionen sind

$$e_k(x) = x^b \bmod N \quad \text{und} \quad d_{k'}(y) = y^a \bmod N$$

Die Verschlüsselungsfunktion e_k heißt auch *RSA-Funktion*, die wir im folgenden auch mit $f_{N,b}$ bezeichnen.

Wir zeigen zunächst, daß e_k und $d_{k'}$ zueinander inverse Ver- und Entschlüsselungsfunktionen sind.

Lemma 9.2. $d_{k'}(e_k(x)) = x$ *für alle* $x \in \mathbb{Z}_N$.

Beweis. ... $\square$

Bei der Berechnung der Funktionen e_k und $d_{k'}$ müssen wir beachten, daß der Wert von a und b die Größenordnung von N haben kann, d.h. a und b können exponentiell in der Größe der Binärdarstellung von N sein. Es gibt aber eine einfache Methode, z.B. $y^a \bmod N$ durch sogenanntes "iteriertes Quadrieren" effizient, d.h. in Zeit polynomiell in $n = \log N$ zu berechnen. Hierzu benützen wir die Binärdarstellung von $a = \sum 2^i a_i$ und die einfache Beziehung $y^{\sum 2^i a_i} = \prod y^{2^i a_i}$. Der Wert von $y^a \bmod N$ ist also das Produkt der Potenzen $y^{2^i} \bmod N$, für die das i-te Bit a_i der Binärdarstellung von a gleich 1 ist. Damit genügt es, schrittweise die Potenzen $y \bmod N, y^2 \bmod N, \ldots, y^{2^{\lfloor a \rfloor}} \bmod N$ zu bestimmen, wobei jede Potenz durch Quadrieren der unmittelbar vorhergenden Potenz berechnet werden kann. Hierfür sind nur noch polynomiell in n viele Multiplikationen nötig, und somit sind e_k und $d_{k'}$ effizient berechenbar.

Die Sicherheit des Verfahrens beruht auf der folgenden *RSA-Annahme*.

RSA-Annahme. Für alle probabilistischen Polynomialzeitalgorithmen E, für alle Polynome s und für genügend große Primzahlen p und q, und für alle $b > 1$ mit $\mathrm{ggt}(b, \varphi(N)) = 1$, wobei $N = pq$, gilt für eine zufällig gewählte Zahl $x \in_{\mathcal{U}} \mathbb{Z}_N$,

$$\Pr\left[E(N, b, f_{N,b}(x)) = x\right] \leq \frac{1}{s(n)},$$

wobei $n = \lceil \log N \rceil$ und $f_{N,b}(x) = x^b \bmod N$ die RSA-Funktion ist.

9.3 Nicht-Erlernbarkeit

Die RSA-Annahme besagt, daß die Umkehrfunktion $f_{N,b}^{-1}$ schwer zu berechnen ist (falls nur der öffentliche Schlüssel $k = (N, b)$ bekannt ist). Die Berechnung von $f_{N,b}^{-1}$ wird nicht wesentlich leichter, wenn wir zusätzlich zufällige Beispiele $(y, f_{N,b}^{-1}(y))$ mit $y \in_U \mathbb{Z}_N$ zur Verfügung haben. Dies liegt daran, daß solche Beispiele die gleiche Verteilung wie Beispiele der Form $(f_{N,b}(x), x)$ mit $x \in_U \mathbb{Z}_N$ besitzen, und deshalb durch zufällige Wahl von $x \in_U \mathbb{Z}_N$

und Berechnung von $f_{N,b}(x)$ effizient generieren werden können. Die Berechnung von $f_{N,b}^{-1}$ mit der Hilfe von Beispielen $(y, f_{N,b}^{-1}(y))$ entspricht aber gerade der Aufgabe, die Umkehrfunktion $f_{N,b}^{-1}$ (in einem erweiterten Sinne) zu "erlernen". Also können wir aus der RSA-Annahme schließen, dass diese Lernaufgabe schwierig ist. Für die entsprechende Definition einer schwierigen Konzeptklasse müssen wir nur noch beachten, daß die Funktionen $f_{N,b}^{-1}$ den Wertebereich $\mathbb{Z}_N$ habe, und daher nicht unmittelbar als Konzepte benutzt werden können.

Sei die Konzeptklasse $C = \bigcup C_n$ mit $C \subseteq \mathcal{P}(\{0,1\}^n)$ wie folgt definiert. Jedes C_n besteht aus allen Konzepten

$$c_{p,q,b,i} = \left\{ y \in \mathbb{Z}_N \mid \text{das } i\text{-te Bit von } f_{N,b}^{-1}(y) \text{ ist } 1 \right\},$$

wobei p und q Primzahlen mit $N = pq \leq 2^n$ sind, $b > 1$ eine Zahl mit $\mathrm{ggt}(b, \varphi(N)) = 1$ ist, und $1 \leq i \leq n$ gilt. Hierbei können wir wegen $n \geq \lceil \log N \rceil$ annehmen, dass alle Zahlen in $y \in \mathbb{Z}_N$ durch einen Binärstring der Länge n kodiert sind. Für diese Binärdarstellung von $y \in \mathbb{Z}_N$ nehmen wir zusätzlich an, daß daraus nicht nur der Wert von y, sondern auch der Wert von N bestimmt werden kann. Es ist klar, daß $c_{p,q,b,i}(f_{N,b}(x)) = x_i$ für alle Konzepte $c_{p,q,b,i} \in C_n$ und für alle $x \in \mathbb{Z}_N$ gilt, und somit haben alle klassifizierte Beispiele für $c_{p,q,b,i}$ die Form $(f_{N,b}(x), x_i)$ mit $x \in \mathbb{Z}_N$.

Theorem 9.3. *Unter der RSA-Annahme ist C nicht effizient pac-erlernbar (mit beliebigen p-evaluierbaren Hypothesenklassen).*

Beweis. Nehmen wir an, dass C mit einer festen Hypothesenklasse H pac-erlernbar ist, und sei A der Lernalgorithmus, der C mit H erlernt. Für alle Zielkonzepte $c_{p,q,b,i}$ in C_n produziert A bei Eingabe n und $\varepsilon = \delta = \frac{1}{4n}$ und mit zufälligen Beispielen

$$(f_{N,b}(x), x_i), \quad x \in_{\mathcal{U}} \mathbb{Z}_N,$$

mit Wahrscheinlichkeit $\geq 1 - \delta$ eine Hypothese h mit

$$err(h) = \Pr[h(f_{N,b}(x)) \neq x_i] \leq \varepsilon$$

für $x \in_{\mathcal{U}} \mathbb{Z}_N$. Daher ist die gesamte Wahrscheinlichkeit für $h(f_{N,b}(x)) \neq x_i$, bei der wir auch die internen Münzwürfe von A und die zufälligen Beispiele einbeziehen, höchstens $\delta + (1-\delta)\varepsilon \leq 1/2n$.

Wir betrachten nun den folgenden probabilistischen Algorithmus B zur Berechnung von x aus der Eingabe N, b und $y = f_{N,b}(x)$.

1. für alle $i = 1, \ldots, n$ mit $n = \lceil \log N \rceil$, simuliere A mit Eingaben n, $\varepsilon = \delta = \frac{1}{4n}$ und klassifizierten Beispielen $(f_{b,N}(x'), x'_i)$ mit $x' \in_{\mathcal{U}} \mathbb{Z}_N$, bis A jeweils eine Hypothese h_i produziert

2. Ausgabe $h_1(y)h_2(y)\ldots h_n(y)$

Die Wahrscheinlichkeit eines Misserfolgs $h_1(f_{N,b}(x))\ldots h_n(f_{N,b}(x)) \neq x$ für $x \in_{\mathcal{U}} \mathbb{Z}_N$ können wir mit

$$\begin{aligned} \Pr[\exists i : h_i(f_{N,b}(x)) \neq x_i] &\leq \sum_{i=1}^{n} \Pr[h_i(f_{N,b}(x))] \\ &\leq n \cdot \frac{1}{2n} = \frac{1}{2} \end{aligned}$$

abschätzen. Da die Laufzeit von B polynomiell in n beschränkt ist, widerspricht dies der RSA-Annahme. □

Jedes Konzept $c_{p,q,b,i} \in C_n$ kann durch einen Schaltkreis polynomieller Größe (in n) berechnet werden, der den ¿Invertierungsexponenten a für $f_{N,b}$ “fest verdrahtet” hat und mit dessen Hilfe die Umkehrfunktion $f_{N,b}^{-1}(y) = y^a \bmod N$ durch “Iteriertes Quadrieren” berechnet. Somit folgt aus der effizienten Erlernbarkeit von Boolschen Schaltkreisen unmittelbar die effiziente Erlernbarkeit von C, und wir erhalten das folgende Korollar.

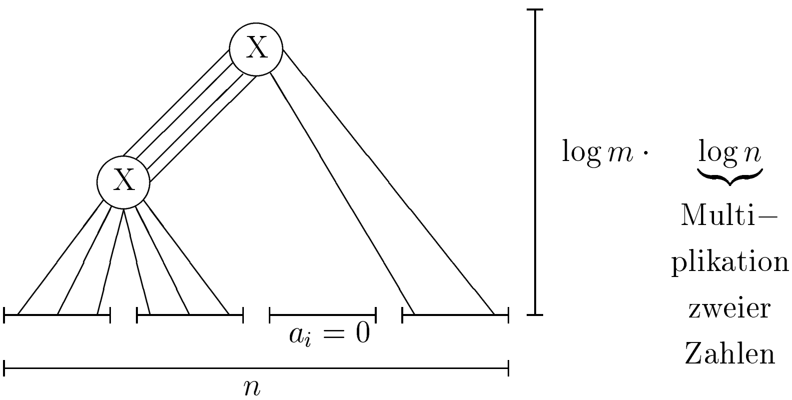
Korollar 9.4. *Unter der RSA-Annahme sind boolsche Schaltkreise nicht mit einer p-evaluierbaren Hypthesenklasse H effizient pac-erlernbar.*

9.4 Nicht-Erlernbarkeit Boolscher Formeln

Eine boolsche Funktion kann genau dann durch eine Boolsche Formel polynomieller Größe dargestellt werden, wenn sie durch einen Schaltkreis logarithmischer Tiefe berechnet werden kann. (Wir betrachten hier nur Schaltkreise mit Fan-In höchstens zwei. Derartige Schaltkreise mit logarithmischer Tiefe können höchstens polynomiell groß sein.)

Die Konzepte der Konzeptklasse C des vorigen Abschnittes waren mit Hilfe der Funktion $f_{N,b}^{-1}(y) = y^a \bmod N$ definiert. Die Berechnung von $y^a \bmod N$ für gegebenes y, a und N durch "iteriertes Quadrieren" erfordert im allgemeinen eine Schaltkreis polynomieller Tiefe, da für die Berechnung einer neuen Potenz $y^{2^i} \bmod N$ der Wert der vorhergehenden Potenz $y^{2^{i-1}} \bmod N$ vorliegen muss. Daher haben die Konzepte in C im allgemeinen keine Schaltkreise logarithmischer Tiefe.

Auf die sequenzielle Berechnung der Potenzen können wir aber verzichten, wenn wir neben y noch zusätzlich die Folge $y \bmod N, y^2 \bmod N, \ldots, y^{2^{\lfloor a \rfloor}} \bmod N$ als Eingabe erhalten. Dann genügt zur Bestimmung von $y^a \bmod N$ die Berechnung des Produkts von höchstens linear vielen Zahlen. Dies führt zunächst einmal zu einem Schaltkreis der Tiefe $O(\log^2 n)$, da bereits die Multiplikation zweier n-bit Zahlen einen Schalkreis der Tiefe $O(\log n)$ benötigt.



Mit Hilfe eines verfeinerten Verfahrens von Beame, Cook und Hoover ist die modulare Multiplikation linear vieler Zahle (iterierte Produkt) mit einem Schaltkreis logarithmischer Tiefe möglich.

Zur Definition der modifizierten Konzeptklasse $C' = \bigcup C'_n$ benützen wir die folgende Notation. Für gegebene N und $y \in \mathbb{Z}_N$ sei

$$pow_N(y) = \left(y \bmod N, y^2 \bmod N, y^4 \bmod N, \ldots, y^{2^{\lceil \log N \rceil}} \bmod N\right).$$

Wie im vorigen Abschnitt gehen wir davon aus, daß die Zahlen $y \in \mathbb{Z}_N$ durch Binärwörter der Länge $n \geq \lceil \log N \rceil$ kodiert sind, so daß aus der Kodierung sowohl y als auch N wiedererlangt werden kann. Aus einer solchen Kodierung können wir dann effizient das Tuppel $pow_N(y)$ berechnen, das wiederum mit einem Binärwort der Länge n^2 kodiert werden kann. Nun definieren wir $C'_n \subseteq \mathcal{P}(\{0,1\}^n)$ als die Menge aller Konzepte

$$c'_{p,q,b,i} = \left\{ pow_N(y) \mid y \in \mathbb{Z}_N \text{ und das } i\text{-te Bit von } f^{-1}_{N,b}(y) \text{ ist } 1 \right\}$$

für Primzahlen p und q mit $N = pq \leq 2^{\lfloor \sqrt{n} \rfloor}$, für $b > 1$ mit $\mathrm{ggt}(b, \varphi(N)) = 1$, und für i mit $1 \leq i \leq n$. Aufgrund der Wahl von N ist sichergestellt, daß es für alle Konzepte $c_{p,q,b,i} \in C_n$ ein Konzept $c'_{p,q,b,i} \in C'_{n^2}$ gibt, so daß

$$c_{p,q,b,i}(y) = c'_{p,q,b,i}(pow_N(y))$$

für alle $y \in \mathbb{Z}_N$ gilt.

Lemma 9.5. *Falls C' mit einer beliebigen p-evaluierbaren Hypothesenklasse H effizient (d.h. Laufzeit polynomiell in n, $\frac{1}{\varepsilon}$ und $\frac{1}{\delta}$) pac-erlernbar ist, so gilt dies auch für C.*

Beweis. Sei A der Lernalgorithmus für C'. Mit Hilfe von A können wir ein Zielkonzept $c_{p,q,b,i} \in C_n$ erlernen, indem wir A mit dem Längenparameter $n' = n^2$ und Zielkonzept $c'_{p,q,b,i} \in C'_{n^2}$ simulieren. Beispiele für A der Form $pow_N(y)$ mit Klassifzierung $c'_{p,q,b,i}(pow_N(y))$ können wir effizient aus angeforderten Beispielen $(y, c_{p,q,b,i}(y))$ generieren. Nach Beendigung der Simulation von A geben wir dann die Hypothese h aus, die durch $h(y) = h'(pow_N(y))$ definiert ist, wobei h' die von A produzierte Hypothese ist. □

Für jedes Konzept in C'_n gibt es einen boolschen Schaltkreis logarithmischer Tiefe, und daher auch eine boolsche Formel. Somit können wir mit Lemma 9.5 und Theorem 9.3 des vorigen Abschnittes unmittelbar auf die Nicht-Erlernbarkeit dieser Konzeptklassen schließen.

Theorem 9.6. *Unter der RSA-Annahme sind die beiden folgenden Konzeptklassen nicht effizient pac-erlernbar (mit beliebigen p-evaluierbaren Hypothesenklassen):*

1. *Schaltkreise logarithmischer Tiefe*
2. *Boolsche Formeln*