Technische Universität Braunschweig

Institut für Theoretische Informatik

Dr. Jürgen Koslowski

Cryptologie 3

Aufgabenblatt 0, 2017-04-15

Übungsaufgabe 1

Beweissystem für quadratische Nichtreste: Das folgende Protokoll ist ein interaktives Beweisstem für die Zugehörigkeit zur Sprache

$$L := \{\, \langle n, x\rangle \in \mathbb{N} \times \mathbb{Z}_n^* \;:\; n = p \cdot q \text{ mit } p\,, q \text{ prim und } x \notin R_n \,\}$$

Gegeben: eine natürliche Zahl n mit (Alice bekannter) Faktorisierung $n = pq$, wobei p und q verschiedenen Primzahlen sind; ein quadratischer Nichtrest $x \in \mathbb{Z}_n^* \backslash R_n$; eine Iterationsschranke $k \in \mathbb{N}$, je nach gewünschter Sicherheit.

Zusammenfassung: Alice beweist Bob, daß x ein quadratischer Nichtrest ist.

for $i := 1$ **to** k **do**

(0) Bob wählt eine Zufallszahl $v \in \mathbb{Z}_n^*$ und berechnet $y = v^2 \bmod n$.

(1) Bob wählt ein zufälliges Bit $i \in \{0, 1\}$ und sendet $z := x^i y \bmod n$ an Alice.

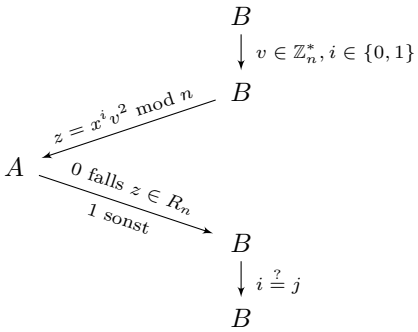
(2) Alice testet z auf Zugehörigkeit zu R_n. Falls ja, sendet sie $j = 0$ an Bob, andernfalls $j = 1$.

(3) Bob prüft, ob $i = j$ gilt.

end;

(4) Bob akzeptiert Alices Beweis, wenn in jeder Runde der Test in Schritt (3) positiv ausfällt.

In Diagrammform:



Aufgrund Ihrer Kenntnis der Faktoren p und q kann Alice z daraufhin überprüfen, ob es sich um einen quadratischen Rest handelt.

(a) [5 PUNKTE] Beweisen Sie, daß dieses Beweissystem vollständig und korrekt ist.

(b) [5 PUNKTE] Beweisen Sie, daß dieses Beweissystem die perfekte Zero-Knowledge-Eigenschaft für Bob hat.

(c) [5 PUNKTE] Zeigen Sie, daß dieses Beweissystem *nicht* die perfekte Zero-Knowledge-Eigenschaft hat.

(d) [5 PUNKTE] Wie läßt sich das obige Protokoll modifizieren, so daß es die perfekte Zero-Knowledge-Eigenschaft doch erfüllt?

Lösungsvorschlag:

Anmerkung: Da die Nichtzugehörigkeit von x zu R_n nicht in offensichtlicher Weise mittels eines Zeugen festgestellt werden kann, handelt es sich beim obigen Protokoll *nicht* um ein Wissensbeweissystem, sondern nur um ein Beweissystem für Sprachzugehörigkeit. Daher stellt sich die Frage nach seiner schwachen Korrektheit (bzw. Validität) nicht.

(a) **Vollständigkeit:** Alice und Bob halten sich an das Protokoll. In Schritt 1 bestimmt Bob einen zufälligen quadratischen Rest $y \in R_n$. Damit ist $z = x^i y \bmod n$ in Schritt 2 im Falle von $i = 0$ ebenfalls quadratischer Rest. Für $i = 1$ handelt es sich bei $z = xy \bmod n$ um einen quadratischen Nichtrest: Nach Satz 9.5 können wir oBdA $x \bmod p \notin R_p$ annehmen. Nach Satz 9.3 erhalten wir dann

$$(xy)^{\frac{p-1}{2}} \bmod p = x^{\frac{p-1}{2}} y^{\frac{p-1}{2}} \bmod p = p - 1$$

woraus wir $(xy) \bmod p \notin R_p$ schließen, und folglich auch $(xy) \bmod n \notin R_n$.

Damit wählt Alice in Schritt 2 immer $j = i$.

Korrektheit: Falls $x \in R_n$ folgt $(xy) \bmod n \in R_n$, denn R_n ist eine Untergruppe von Z_n^*. Damit gilt unabhängig von Bobs Wahl von $i \in \{0, 1\}$ immer $z \in R_n$, und Alice müßte korrekterweise $j = 0$ senden. Unabhängig von ihrer Strategie stimmt in jeder Runde ihr tatsächlich gesendeter Wert j' mit Wahrscheinlichkeit $\frac{1}{2}$ mit Bobs gewähltem i überein. Folglich beträgt die Wahrscheinlichkeit für positive Ergebnisse in Schritt 4 in allen k Runden 2^{-k}.

(b) Achtung: in Definition 11.2 und 11.3 im Buch muß es eigentlich heißen: Ein interaktives Polynomialzeit-Beweissystem hat die *perfekte Zero-Knowledge-Eigenschaft für Bob* bzw. *ohne Qualifizierung*, wenn ein Simulator S bzw. für jede Strategie B^* ein Simulator S^* mit den angegebenen Eigenschaften existiert.

Der Nachweis der entsprechenden Zero-Knowledge-Eigenschaft erfolgt also immer in zwei Schritten:

- Konstruktion eines Simulators
- Nachweis, daß er dieselben Transkripte mit derselben Wahrscheinlichkeitsverteilung erzeugt, wie sie beim tatsächlichen Ablauf des Protokolls auftreten.

Dabei versteht man unter einem Transkript eine Auflistung der bei Protkollausführung ausgetauschten Nachrichten.

Echte Transkripte bestehen aus k Paaren der Form $\langle z, j \rangle \in \mathbb{Z}_n^* \times \{0, 1\}$ mit der Eigenschaft

$$z \in R_n \text{ sofern } j = 0 \quad \text{bzw.} \quad z \in x \cdot R_n \text{ sofern } j = 1$$

Insbesondere können in legalen Paaren *nicht* alle Werte aus $\mathbb{Z}_n^*$ in der ersten Komponente auftreten, sondern nur Elemente aus R_n und genau einer der drei weiteren Nebenklassen, nämlich $x \cdot R_n$. Die Anzahl der legalen Paare ist daher durch die doppelte Größe von R_n gegeben. Aus den Sätzen 9.2 und 9.5 schließen wir $|R_n| = \frac{\varphi(n)}{4}$, also gibt es $\frac{\varphi(n)}{2}$ legale Paare, die alle gleichwahrscheinlich sind.

Wir konstruieren einen Simulator wie folgt:

Eingabe: $x \in \mathbb{Z}_n^* \setminus R_n$

Ausgabe: simuliertes Transkript als Liste L

1: {**(1) Initialisierung**}
2: $L := x$
3: $t := 0$

4: {**(2) Erzeugung der Protokolldaten**}
5: **while** $t < k$ **do**
6: wähle zufällig $v \in \mathbb{Z}_n^*$
7: wähle zufällig $i \in \{0, 1\}$
8: $z := x^i v^2 \bmod n$
9: $L := L \| (z, i)$
10: $t := t + 1$
11: **end while**

Die Menge der simulierten Transkripte für vorgegebenes $x \in \mathbb{Z}_n^* \setminus R_n$ stimmt offenbar mit der Menge der echten Transkripte überein. Und da der Simulator dieselben Berechnungen vornimmt, die auch im echten Protokollablauf erforderlich sind, stimmt i mit j bei korrekter Durchführung immer überein. Folglich haben die echten und die simulierten Transkripte dieselbe Wahrscheinlichkeitsverteilung.

(c) Bob verwendet einen probabilistischen Polynomialzeitalgorithmus B^*, um seine Herausforderung z^* an Alice zu erzeugen. Dieser verwendet die Eingaben $n = pq$ und $x \in \mathbb{Z}_n^* \setminus R_n$ und verfügt in Runde t über einen Zustand Q_t, $p < k$, der variieren kann, indem er z.B. die Ergebnisse der früheren Runden speichert.

Ein potentieller Simulator S^* muß B^* aufrufen und aus der Ausgabe z^* ein gültiges Paar $\langle z^*, i \rangle$ erzeugen.

Der korrekte Wert i für gegebenes z^* kann nicht berechnet werden, da die Überprüfung von $z^* \in R_n$ die Kenntnis der Faktoren p und q erfordert.

Also versuchen wir wie im Beweis von Satz 11.3 in einer Schleife solange gültige Paare $\langle z, i \rangle$ zu erzeugen, bis z mit der von B^* gelieferten Herausforderung z^* übereinstimmt:

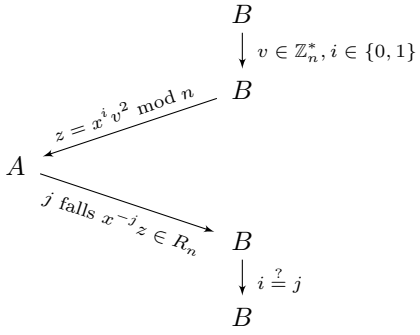
1: {**(1) Initialisierung**}
2: $L := x$
3: $t := 0$
4: {**(2) Simulierung der Protokolldaten**}
5: **while** $t < k$ **do**
6: $Q_t :=$ Zustand von B^*
7: $z^* := B^*(n, x, Q_t)$
8: **repeat**
9: wähle zufällig $v \in \mathbb{Z}_n^*$
10: wähle zufällig $i \in \{0, 1\}$
11: $z := x^i v^2 \bmod n$
12: **until** $z = z^*$
13: $L := L \| (z, i)$
14: $t := t + 1$
15: **end while**

Das Problem dieses und jedes vergleichbaren Simulators ist, daß er $z^* \in R_n + x \cdot R_n$ voraussetzt. Aber warum sollte sich ein unehrlicher Bob an diese Bedinung halten müssen? Er kann genausogut Challenges aus den anderen beiden Nebenklassen von R_n wählen, die weder quadratische Reste, noch Produkt von x mit einem quadratischen Rest sind.

Im obigen Protokoll muß Alice in diesem Fall mit 1 antworten, aber $\langle z^*, 1 \rangle$ ist kein legales Paar im Sinne von (b).

(d) Alice kann (und sollte) Bobs Challenge z darauf testen, ob er zu $R_n + x \cdot R_n$ gehört. Falls $z \notin R_n$, muß sie nur überprüfen, ob $x^{-1} z \bmod n \in R_n$ gilt. Nur in diesem Fall sollte sie die Antwort 1 schicken, ansonsten weiß sie, daß Bob versucht hat, zu betrügen.

In Diagrammform:



Dies schränkt Bobs Betrugsmöglichkeit darauf ein, die Häufigkeiten der Challenges aus R_n bzw. aus $x \cdot R_n$ unterschiedlich zu gestalten, garantiert aber in jedem Durchlauf ein legales Paar.

Bei Ablauf des echten Protokolls tritt ein gültiges Paar $\langle z^*, j\rangle$ in Durchlauf t mit derselben Wahrscheinlichkeit auf, mit der $B^*(n, x, Q_t)$ die Ausgabe z^* liefert; wir bezeichnen diese mit $P^*(z^*, Q_t)$.

Nun suchen wir die Wahrscheinlichkeit $P(t, z^*, j)$, daß der obige Simulator S^* im Schritt $t < k$ das gültige Paar $\langle z^*, j\rangle$ an die Liste L anhängt. In Schritt 6 wird mit derselben Wahrscheinlichkeit wie im echten Transkript der Wert z^* bestimmt. Die folgende Schleife dient nur dazu, den dazu gehörigen korrekten Wert j zu finden, dieser hat keinen weiteren Einfluß auf die Wahrscheinlichkeit! Daraus folgt schon die unqualifizierte perfekte Zero-Knowledge-Eigenschaft.

Wir betrachten trotzdem die k Runden einzeln. In den Schritten (8)-(10) wird jedes gültiges Paar $\langle z, j\rangle$ mit der Wahrscheinlichkeit $\frac{2}{\varphi(n)}$ gewählt. Anschließend liefert B^* mit Wahrscheinichkeit $P^*(z^*, Q_t)$ die Herausforderung z^*. Die Wahrscheinlichkeit für $z^* = z$ beträgt also $P(t, z, z^*) = \frac{2P^*(z^*, Q_t)}{\varphi(n)}$. Wenn $t > 0$ Iterationen benötigt werden, bis erstmals $z^* = z$ gilt, ergibt sich die Wahrscheinlichkeit, ein Paar mit erster Komponente z^* an die Liste anzuhängen, zu

$$P^*(z^*, Q_t) \sum_{t=0}^{\infty} \big(1 - P(p, z, z^*)\big)^t P(p, z, z^*) = P^*(z^*, Q_t) P(p, z, z^*) \frac{1}{P(p, z, z^*)} = P^*(z^*, Q_t)$$

genau wie beim echten Transkript.

Wir müssen noch zeigen, daß die erwartete Laufzeit von S^* polynomial in n und k ist. Der Erwartungswert für die Anzahl der Durchläufe der inneren Schleife ist gegeben durch

$$T = P(t, z, z^*) \sum_{t=0}^{\infty} (t+1)\big(1 - P(t, z, z^*)\big)^t$$

Mit der Abkürzung $Q := 1 - P(t, z, z^*)$ vereinfacht sie die Summe zu

$$S = \sum_{t=0}^{\infty} tQ^t + \sum_{t=0}^{\infty} Q^t = Q \sum_{t=1}^{\infty} tQ^{t-1} + \frac{1}{1-Q} = QS + \frac{1}{1-Q}$$

woraus wir schließen

$$S = \frac{1}{(1-Q)^2} \quad \text{und damit} \quad T = \frac{1}{P(t, z, z^*)} = \frac{\varphi(n)}{2P^*(z^*, Q_t)}$$

Dabei ist $P^*(z^*, Q_t)$ eine positive Konstante, da der Wert z^* tatsächlich aufgetreten ist.

Die erwartete Laufzeit von S^* ist folglich von der Ordnung $O(k \cdot \varphi(n))$, was polynomial in den Werten k und n ist (aber exponentiell in den Codierungen von k und n, sofern diese nicht unär erfolgt).