Laborpraktikum Verteilte Systeme
Sommersemester 2017

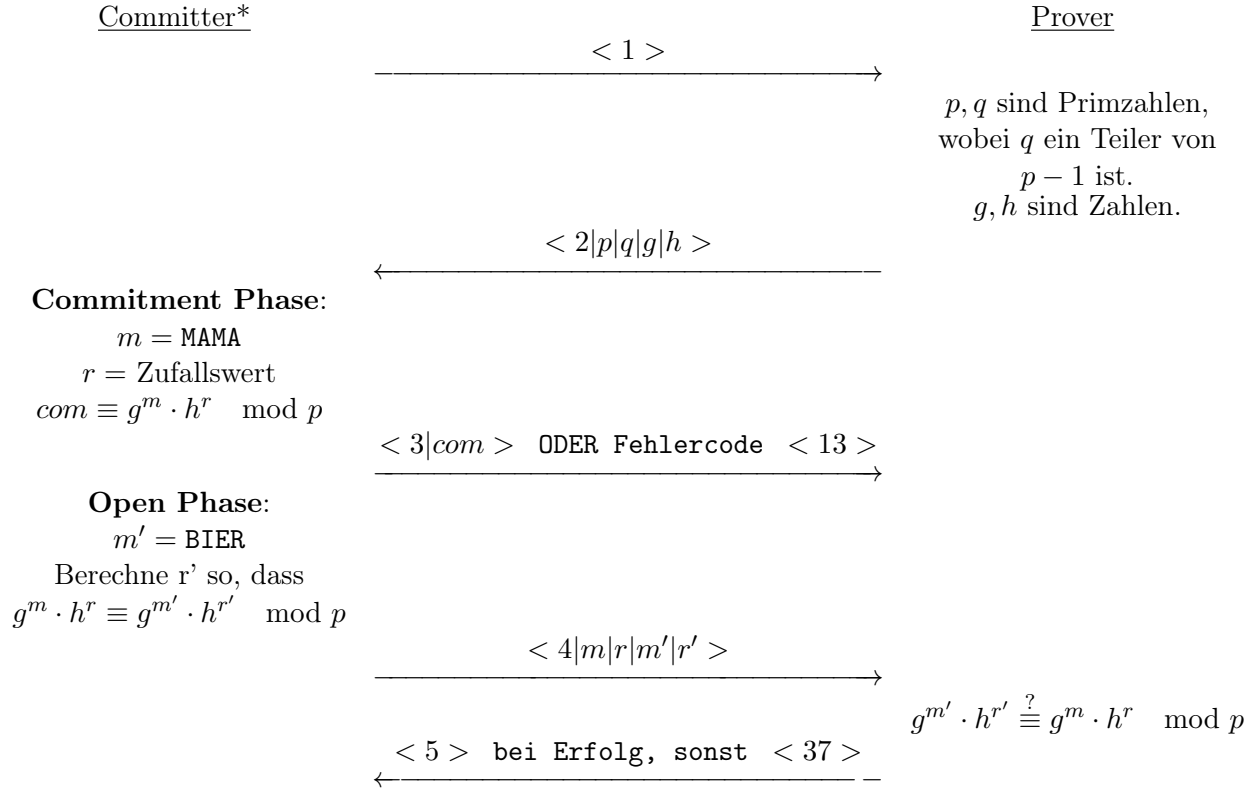
Prof. Dr.-Ing. Andreas Noack
Fachhochschule Stralsund

1 Netzwerkprotokoll: Commitment Schema

Ein Commitment Schema ist ein kryptographisches Experiment, bei dem der Committer sich auf eine Information m festlegt und einen Nachweis für seine Informationswahl (com) erzeugt, ohne jedoch etwas über die Information zu verraten (Commitment Phase). Zu einem späteren Zeitpunkt legt er seine Information offen (Open Phase). Dabei gibt es zwei wichtige Eigenschaften:

- **Hiding** – Die Information m bleibt bis zur Open Phase vertraulich.
- **Binding** – Der Committer kann seine Information nach der Commitment Phase nicht mehr ändern.

Die Protokollabbildung 1 zeigt die Realisierung des Experiments als Netzwerkprotokoll.



Protokoll 1: Bit Commitment mit diskretem Logarithmus

Anmerkung: *p, q, g, h, com, r, r'* sind jeweils sehr große Zahlen flexibler Länge. Jede dieser Zahlen wird in dem Format $< Länge_{[2\ Bytes]} \mid Zahl_{[Länge\ Bytes]} >$ übertragen. Zahlenkonstanten werden als einzelne Bytes übertragen, m und m' (ASCII) mit jeweils 4 Bytes. Zur Codierung wird Big Endian verwendet.

2 Aufgabe

Entwickeln Sie ein Programm, das die Rolle des Committers übernimmt und die Binding-Eigenschaft des Protokolls brechen kann.

Zu diesem Zweck müssen Sie zunächst den diskreten Logarithmus $log_h(g) \equiv x \mod p$ lösen, d.h. $h^? \equiv g \mod p$. Dies kann via Bruteforce oder effizienteren Algorithmen wie *Baby Step Giant Step* gelöst werden. Hierbei kommt Ihnen zur Hilfe, dass die Untergruppe $\boldsymbol{Z}_q$ aus Versehen unsicher gewählt wurde, d.h. die Anzahl der Element in dieser Gruppe entspricht einem im Labor berechenbaren Sicherheitsniveau.

Sobald x bekannt ist, kann mit der folgenden Idee der benötigte Wert r' berechnet werden:

$$
\begin{aligned}
& g^m \cdot h^r && \equiv g^{m'} \cdot h^{r'} \mod p \\
\Leftrightarrow & g^{m-m'} && \equiv h^{r'-r} \mod p \\
\Leftrightarrow & g && \equiv h^{(r'-r)\cdot(m-m')^{-1}} \mod p \\
\overset{(\text{mit } g \equiv h^x \mod p)}{\rightarrow} & h^x && \equiv h^{(r'-r)\cdot(m-m')^{-1}} \mod p \\
\Leftrightarrow & x && \equiv (r'-r)\cdot(m-m')^{-1} \mod q \\
\Leftrightarrow & x \cdot (m-m') + r && \equiv r' \mod q
\end{aligned}
$$

Erschwerend kommt hinzu, dass der Prover bösartig implementiert ist. Dies bedeutet, dass er manchmal nicht plausible Parameter oder einfach nur falsche Netzwerknachrichten erzeugt. Tritt so ein Fall auf, soll Ihr Programm dies erkennen und mit einer entsprechenden Fehlermeldung (Fehlercode 13) quittieren!

Diese Aufgabe kann mit verschiedenen Schwierigkeitsgraden gelöst werden, die in die spätere Benotung einfließen. Eine Einordnung der Schwierigkeitsgrade sieht wie folgt aus:

1. Keine Robustheit gegen Fehler, einfacher Algorithmus zur Lösung des DLP
2. Mehr als 50% Robustheit gegen Fehler, einfacher Algorithmus zur Lösung des DLP
3. Mehr als 95% Robustheit gegen Fehler, einfacher Algorithmus zur Lösung des DLP
4. Mehr als 95% Robustheit gegen Fehler, effizienter Algorithmus zur Lösung des DLP

Die **Wahl der Programmiersprache** und Bibliotheken wird nur durch die Verfügbarkeit auf den Linux-Systemen im Labor eingeschränkt.

1. **Abgabe bis 24.3.2016 (23:59 Uhr)** per E-Mail.
   - Wählen Sie eine Programmiersprache aus.
   - Erarbeiten Sie die theoretischen Grundlagen (mathematisches Problem, Netzwerkgrundlagen)
   - Erarbeiten Sie ein Konzept für Ihr Programm, d.h. welche Bibiotheken/Techniken wollen Sie verwenden und wie lauten die Befehle für diese Vorgehensweise?

2. **Abgabe bis 13.5.2016 (23:59)** per E-Mail.
   - Erstellen Sie eine erste lauffähige Implementierung, die wenigstens folgende Eigenschaften erfüllt:
     - Erfolgreicher Protokolldurchlauf mit einem ehrlichen Prover
     - Berechnung eines gültigen $< m, r, m', r' >$-Tupels
3. **Abgabe bis 2.6.2016 (23:59)** per E-Mail.
   - Erstellen Sie eine lauffähige Implementierung, die wenigstens folgende Eigenschaften erfüllt:
     - Automatisierte Kommunikation mit dem Prover (z.B. 100 Durchläufe)
     - Wenn der Prover ehrlich spielt, Berechnung eines gültigen $< m, r, m', r' >$-Tupels
4. **Abgabe bis 16.6.2017 (23:59)** per E-Mail.
   - Optimieren Sie Ihre Lösung hinsichtlich der Performance!
   - Finale Abgabe!
5. **Abschlusspräsentation am 21.6.2017**
   Zusätzlich zu der Implementierung wird eine Abschlusspräsentation (20 Min) erwartet, in der die eigene Entwicklung und die Hintergründe erläutert werden.
   Die schnellste Implementierung wird prämiert.

Hinweis: *Am Mittwoch nach jeder Abgabe (29.3., 17.5., 7.6., 21.6.) findet ein* ***Präsenztermin*** *im Labor 4/221 statt. Weitere Termine nach Absprache.*

Bei Fragen können Sie mich gerne kontaktieren:

Prof. Dr.-Ing. Andreas Noack
Fachhochschule Stralsund
Zur Schwedenschanze 15
18435 Stralsund
03831/45-6626
andreas.noack@fh-stralsund.de