Institut für Theoretische Informatik
Prof. Dr. Dennis Hofheinz
Übungsleiter: Thomas Agrikola

Stammvorlesung Sicherheit im Sommersemester 2017

Übungsblatt 6

Aufgabe 1. Der ebenso geniale wie misstrauische Wissenschaftler und Superbösewicht Doktor Meta hat aus der Vergangenheit gelernt. Als er noch, der Tradition folgend, seine unkonkreten Pläne offen kommunizierte, fiel es seinen Widersachern leicht, sie zu durchkreuzen. Er möchte in der Zukunft diskreter arbeiten.

Da er neuerdings seine Handlanger extern von der Agentur „Bösewicht Consulting Group“ bezieht, möchte er deren Arbeit konkreter überwachen. Dazu möchte er das Chinese-Wall-Modell in seinem Computernetzwerk umsetzen. Greift ein Handlanger bei der Arbeit in einer von Doktor Metas Scheinfirmen auf ein Dokument einer gesperrten Scheinfirma zu, wird er in das neu erworbene Haifischbecken versetzt.

Gegeben seien

- die Scheinfirmen $\mathcal{C} = \{c_1, c_2, c_3\}$ mit
  $c_1 =$ Tracebook, $c_2 =$ Los Pollos HerMetas und $c_3 =$ MetaMarkt,
- die Berater $\mathcal{S} = \{s_1, s_2, s_3\}$ mit
  $s_1 =$ Dr. Neurocide, $s_2 =$ Montezuma, $s_3 =$ Red Ivan, und
- die Objekte $\mathcal{O} = \{o_1, o_2, o_3\}$ mit
  $o_1 =$ Inventarliste, $o_2 =$ Liste der geschmierten Politiker, $o_3 =$ Lageplan Untergrundlabore.

Es sei $y(o_i) = c_i$, für $i \in \{1, 2, 3\}$, $x(o_1) = \emptyset$, $x(o_2) = \{c_1, c_3\}$ und $x(o_3) = \{c_1\}$. Betrachten Sie die folgende Abfolge von Anfragen $b \in \mathcal{S} \times \mathcal{O}$ in Reihenfolge:

1. (s_3, o_3) (`read`)
2. (s_3, o_1) (`read`)
3. (s_3, o_2) (`write`)
4. (s_2, o_1) (`read`)
5. (s_1, o_2) (`write`)
6. (s_1, o_1) (`read`)
7. (s_2, o_2) (`write`)
8. (s_3, o_3) (`write`)
9. (s_2, o_3) (`read`)
10. (s_1, o_1) (`write`)

Beschreiben Sie, ob die einzelnen Anfragen gewährt werden. Falls nicht, welche Eigenschaft(en) – im Sinne der ss- oder $\star$-Eigenschaft – wurde(n) verletzt?

Lösungsvorschlag zu Aufgabe 1.

	Anfrage	ss	$\star$	Begründung
1.	(s_3, o_3) (`read`)	✓	✓	–
2.	(s_3, o_1) (`read`)	×	✓	s_3 liest schon o_3 und $y(o_1) \in x(o_3)$
3.	(s_3, o_2) (`write`)	✓	×	s_3 liest schon o_3 und $x(o_3) \neq \emptyset$
4.	(s_2, o_1) (`read`)	✓	✓	–
5.	(s_1, o_2) (`write`)	✓	✓	–
6.	(s_1, o_1) (`read`)	×	✓	s_1 hat schon Zugriff auf o_2 und $y(o_1) \in x(o_2)$
7.	(s_2, o_2) (`write`)	✓	✓	(s_2 liest schon o_1, aber $x(o_1) = \emptyset$)
8.	(s_3, o_3) (`write`)	✓	✓	–
9.	(s_2, o_3) (`read`)	×	✓	s_2 hat schon Zugriff auf o_2 und $y(o_3) \in x(o_2)$
10.	(s_1, o_1) (`write`)	×	✓	s_1 hat schon Zugriff auf o_2 und $y(o_1) \in x(o_2)$

Aufgabe 2. Gegeben sei das folgende System im Bell-LaPadula-Modell:

- Subjektmenge $\mathcal{S} = \{\text{Alice}, \text{Bob}, \text{Carol}\}$
- Objektmenge $\mathcal{O} = \{D_1, D_2, D_3, D_4\}$
- Menge der Zugriffsoperationen $\mathcal{A} = \{\texttt{read}, \texttt{write}, \texttt{append}, \texttt{execute}\}$
- Menge der Sicherheitsstufen $L = \{\mathit{Verwaltung}, \mathit{Lehre}, \mathit{Forschung}, \mathit{Praesidium}\}$ mit der partiellen Ordnung definiert durch $\mathit{Lehre} \leq \mathit{Verwaltung} \leq \mathit{Praesidium}$ und $\mathit{Lehre} \leq \mathit{Forschung} \leq \mathit{Praesidium}$
- Zugriffskontrollmatrix M gegeben durch

	D_1	D_2	D_3	D_4
Alice	`r,w,a`	`r`	`r,w,a`	`r,x`
Bob	`r,w,a`	`r,w,a`	`r,w,a`	`r,x`
Carol	`r`	`r`	`r,w,a`	`r,w,a,x`

- Zuordnung der Sicherheitsstufen $F = (f_S, f_C, f_O)$ gegeben durch

	f_S	f_C
Alice	*Verwaltung*	*Verwaltung*
Bob	*Forschung*	*Lehre*
Carol	*Praesidium*	*Forschung*

	f_O
D_1	*Verwaltung*
D_2	*Lehre*
D_3	*Forschung*
D_4	*Praesidium*

Geben Sie für die folgende Liste von Zugriffsanforderungen und Anfragen zur Änderung der Sicherheitsstufe an, ob das System Zugriff gewährt beziehungsweise die Sicherheitsstufe ändert oder nicht. Berücksichtigen Sie erfolgreiche Zugriffe und Änderungen der Sicherheitsstufe für die nachfolgenden Zugriffsanforderungen. Geben Sie für abgelehnte Anfragen als Begründung an, welche Eigenschaft(en) verletzt würde(n), wenn der Zugriff beziehungsweise die Änderung der Sicherheitsstufe erlaubt würde. Dabei bezeichne (s, o, l) eine Zugriffsanforderung von Subjekt s auf Objekt o mit der Zugriffsart l. Gehen Sie von einem leeren Initialzustand aus.

1. $(\mathit{Alice}, D_1, \texttt{a})$
2. $(\mathit{Bob}, D_2, \texttt{w})$
3. $(\mathit{Bob}, D_3, \texttt{r})$
4. $(\mathit{Carol}, D_3, \texttt{r})$
5. $(\mathit{Carol}, D_2, \texttt{w})$
6. $(\mathit{Alice}, D_2, \texttt{r})$
7. $(\mathit{Bob}, D_4, \texttt{r})$
8. $(\mathit{Bob}, D_2, \texttt{a})$

Lösungsvorschlag zu Aufgabe 2.

	Anforderung	Zugriff erteilt/verweigert
1.	$(Alice, D_1, \mathtt{a})$	erteilt
2.	$(Bob, D_2, \mathtt{w})$	erteilt
3.	$(Bob, D_3, \mathtt{r})$	erteilt, $f_C(Bob) = f_O(D_3) =$ Forschung
4.	$(Carol, D_3, \mathtt{r})$	erteilt, $f_C(Carol)$ bleibt unverändert
5.	$(Carol, D_2, \mathtt{w})$	verweigert ($*$- & ds-Eigenschaft)
6.	$(Alice, D_2, \mathtt{r})$	erteilt, $f_C(Alice)$ bleibt unverändert
7.	$(Bob, D_4, \mathtt{r})$	verweigert (ss-Eigenschaft)
8.	$(Bob, D_2, \mathtt{a})$	verweigert ($*$-Eigenschaft)

Aufgabe 3. Wir betrachten ein Public-Key-Identifikationsprotokoll $(\mathsf{Gen}, \mathsf{P}, \mathsf{V})$, bei dem ein Benutzer P einen Benutzer V davon überzeugen will, dass P einen diskreten Logarithmus $\log_g h$ modulo p, mit ungeradem primem $p \in \mathbb{N}$ und $g, h \in \mathbb{Z}_p^\times$, kennt. Wir nehmen dazu an, dass beide Benutzer dem Protokoll folgen. Dabei sei der Beweisalgorithmus P, der Verifikationsalgoritmus V sowie der Protokollablauf folgendermaßen definiert:

(1) Die Parametergenerierung $\mathsf{Gen}(1^k)$ wählt zuerst ein ungerades primes $q \in \mathbb{N}$ und setzt $p := 2q + 1$, sodass p prim ist. Weiterhin sei $g \in \mathbb{Z}_p^\times$ ein Element der Ordnung $q := \mathsf{ord}(g)$. Wir ziehen zufällig gleichverteilt ein $s \in \mathbb{Z}_{\mathsf{ord}(g)}$ und setzen $h := g^s \bmod p$. ($\mathsf{ord}(g)$ gebe dabei die Ordnung von g in $\mathbb{Z}_p^\times$ an.) Anschließend wird ein Public-Key $pk := (\mathbb{Z}_p^\times, g, h)$ und ein Secret-Key $sk := (\mathbb{Z}_p^\times, g, s)$ ausgegeben.

(2) $\mathsf{P}(sk)$ wird mit Eingabe des Secret-Keys sk gestartet, wählt ein zufällig gleichverteiltes $r \in \mathbb{Z}_{\mathsf{ord}(g)}$ und gibt $\mathsf{out}_\mathsf{P} := X := g^r \bmod p$ aus.

(3) Der Verifikationsalgorithmus $\mathsf{V}(pk, \mathsf{out}_\mathsf{P})$ erhält als Eingabe den Public-Key pk und die P-Ausgabe $X := \mathsf{out}_\mathsf{P}$ aus (2); anschließend wird ein $b \in \{0, 1\}$ zufällig gleichverteilt gezogen und $\mathsf{out}_\mathsf{V} := b$ ausgegeben.

(4) $\mathsf{P}(sk, \mathsf{out}_\mathsf{V})$ erhält zu sk die V-Ausgabe $b := \mathsf{out}_\mathsf{V}$ aus (3), berechnet $y := r + b \cdot s \bmod \mathsf{ord}(g)$ und gibt $\mathsf{out}_\mathsf{P} := y$ aus.

(5) $\mathsf{V}(pk, \mathsf{out}_\mathsf{P})$ erhält zu pk die P-Ausgabe $y := \mathsf{out}_\mathsf{V}$ aus (4), überprüft, ob $g^y \bmod p = h^b X \bmod p$ gilt. Falls ja, wird 1 ausgegeben; anderenfalls 0.

Wir sagen, dass V den Beweis von P akzeptiert, wenn nach (mehrfacher) Anwendung des Protokolls immer $\mathsf{V}(pk, \mathsf{out}_\mathsf{P}) = 1$ in (5) gilt.

(a) Zeigen Sie die Korrektheit von $(\mathsf{Gen}, \mathsf{P}, \mathsf{V})$.

(b) Falls V in (3) $b = 0$ zieht, ist y in (4) offensichtlich unabhängig von s. (Das bedeutet, dass jemand ohne Kenntnis von sk in (2) $g^r \bmod p$ und in (4) r senden würde, sodass V akzeptiert.) Falls V in (3) $b = 1$ wählt, ist dies nicht der Fall. Wie könnte jemand, der sk nicht kennt, in diesem Fall dennoch Elemente X in (2) und y in (4) erzeugen, sodass V in (5) 1 ausgibt? Wie groß ist die Erfolgswahrscheinlichkeit?

(c) Geben Sie einen Simulator $\mathcal{S}$ in der Rolle von P (analog zum Graph-Dreifärbbarkeitsbeispiel aus der Vorlesung) an.

Lösungsvorschlag zu Aufgabe 3.

(a) Für $(pk, sk) := \mathsf{Gen}(1^k)$, mit pk und sk wie in (1), $X := \mathsf{P}(sk)$ in (2), $b := \mathsf{V}(pk, X)$ in (3) und $y := \mathsf{P}(sk, b)$ in (4) gilt

$$g^y \bmod p = g^{r+b\cdot s} \bmod p = h^b X \bmod p$$

und damit $\mathsf{V}(pk, \mathsf{out}_\mathsf{P}^{(4)}) = \mathsf{yes}$.

(b) Wir wählen in (2) ein Bit $b' \in \{0,1\}$ zufällig gleichverteilt, setzen $X := g^r h^{-b'} \bmod p$, für zufällig gleichverteiltes $r \in \mathbb{Z}_{\mathsf{ord}(g)}$, und senden X an V. (Hinweis: Der Public-Key $pk = (\mathbb{Z}_p^\times, g, h)$ ist uns bekannt. Da $g^r \bmod p$ ein zufälliges Element in $\mathbb{Z}_p^\times$ ist, ist auch $g^r h^{-b'} \bmod p$ ein zufälliges Element in $\mathbb{Z}_p^\times$. Das heißt, der Wert X in (3) sieht in beiden Fällen, $b' = 0$ oder $b' = 1$, zufällig für V aus. Vs Wahl von b in (3) ist also unabhängig von b'.) Mit Wahrscheinlichkeit $\frac{1}{2}$ haben wir $b' = b$ gezogen. Sendet V uns in (3) ein Bit $b = b'$, setzen wir in (4) $y := r$ und senden dieses an V. V verifiziert

$$g^y \bmod p = h^b X \bmod p = h^b g^r h^{-b'} \bmod p = g^r \bmod p$$

und gibt yes aus. Falls $b' \neq b$ gilt, brechen wir ab. Mit Wahrscheinlichkeit $\frac{1}{2}$ gibt V in (5) den Wert yes aus.

(c) Wir geben einen Simulator $\mathcal{S}$ an, der ein Transkript (X, b, y) ausgibt, sodass die Verteilungen $\langle \mathsf{P}(sk), \mathcal{A}(1^k, pk)\rangle$ mit Angreifer $\mathcal{A}$ und (die Ausgabe von) $\mathcal{S}(1^k, pk)$ ununterscheidbar sind. Der Simulator $\mathcal{S}(1^k, pk)$ agiere wie folgt beschrieben:

(i) $\mathcal{S}$ spielt Protokoll mit $\mathcal{A}$, in der Rolle von P.

(ii) $\mathcal{S}$ sendet $X := g^r h^{-b'} \bmod p$, für zufällig gleichverteiltes $r \in \mathbb{Z}_{\mathsf{ord}(g)}$ und zufällig gleichverteiltes $b' \in \{0,1\}$, an $\mathcal{A}$.

(iii) $\mathcal{S}$ hofft auf $b' = b$, nachdem $\mathcal{A}$ ein Bit $b \in \{0,1\}$ gesendet hat. Ist dies der Fall, wird das Protokoll fortgeführt. Anderenfalls, spult $\mathcal{S}$ den Angreifer $\mathcal{A}$ zurück und spielt das Spiel von vorn.

(iv) $\mathcal{S}$ sendet den Wert $y := r$ aus (ii) an $\mathcal{A}$. (Dies passiert, falls $\mathcal{A}$ in (iii) nicht zurückgespult wurde.)

(v) $\mathcal{S}$ gibt schließlich Transkript (X, b, y) aus. (Es gilt $g^y \bmod p = h^b X \bmod p$.)

Wie wir in (b) sahen, ist $\mathcal{A}$s Wahl von b in (iii) unabhängig von b'. Wir erhalten für die Wahrscheinlichkeit, dass $b' = b$ gilt, $\frac{1}{2}$. Damit ist die Wahrscheinlichkeit, dass der Simulator k Versuche braucht, bis er terminiert, 2^{-k}. Die erwartete Laufzeit ist dann $T(n) \sum_{k=1}^{\infty} 2^{-k} = O(T(n))$, wobei $T(n)$ der Laufzeit eines Durchlaufs, inklusive der Simulation von $\mathcal{A}$ entspricht. Da $T(n)$ polynomiell ist, ist die erwartete Laufzeit des Simulators polynomiell.

Wir zeigen nun, dass die Ausgabeverteilung von $(pk, \langle \mathsf{P}(sk), \mathcal{A}(1^k, pk)\rangle)$ und $(pk, \mathcal{S}(1^k, pk))$ ununterscheidbar sind. Ein ehrliches V kann also nicht (effizient) sagen, ob V mit einem ehrlichen P oder mit $\mathcal{S}$ das Protokoll führt. Wir wissen aus (b), dass X für $b' = 0$ oder $b' = 1$ zufällig verteilt ist. Sei nun $b := \mathsf{V}(pk, X)$, konditioniert auf $b' = b$. Wie oben gesehen, ist die Wahrscheinlichkeit dafür $\frac{1}{2}$. Für einen Beweiser P sowie für einen Simulator $\mathcal{S}$ ist für $b = 0$ der Wert y ein zufälliger Wert aus $\mathbb{Z}_{\mathsf{ord}(g)}$; für $b = 1$ ebenso.

Aufgabe 4. Wir betrachten im Folgenden das Jacobi-Symobol $\left(\frac{a}{n}\right)$ für zwei natürliche Zahlen $a, n \in \mathbb{N}$. Dabei gilt, falls $n = p \in \mathbb{P}$ eine Primzahl ist, folgendes:

$$\left(\frac{a}{p}\right) := a^{\frac{p-1}{2}} \bmod p = \begin{cases} 1 & \text{wenn } a \text{ quadratischer Rest modulo } p \text{ ist} \\ -1 & \text{wenn } a \text{ quadratischer Nichtrest modulo } p \text{ ist} \\ 0 & \text{wenn } a \text{ ein Vielfaches von } p \text{ ist} \end{cases}$$

Desweiteren gelten folgende Rechenregeln für ungerades $n \in \mathbb{N}$:

* $\left(\frac{a}{n}\right) = \left(\frac{a \bmod n}{n}\right), \left(\frac{-1}{n}\right) = (-1)^{\frac{n-1}{2}}$
* $\left(\frac{a_1 a_2}{n}\right) = \left(\frac{a_1}{n}\right)\left(\frac{a_2}{n}\right), \left(\frac{a}{mn}\right) = \left(\frac{a}{m}\right)\left(\frac{a}{n}\right)$

(a) Berechnen Sie die Jacobi-Symbole von $\left(\frac{15}{35}\right)$, $\left(\frac{32}{33}\right)$, $\left(\frac{17}{39}\right)$

(b) Zeigen Sie, dass -1 für $n = pq$ mit Primzahlen p und q für die $p \equiv q \equiv 3 \bmod 4$ gilt, kein quadratischer Rest mod n ist, jedoch das Jacobi-Symbol 1 ergibt. (Solche Zahlen n nennt man Blum-Integer.)

(c) Wir betrachten nun folgendes Identifikationsprotokoll, das auf der Schwierigkeit der Berechnung von Quadratwurzeln in der Gruppe $\mathbb{Z}_n^\times$ für $n = pq$ Blum-Integer besteht, sofern n nicht leicht zu faktorisieren ist:

Eine vertrauenswürdige Instanz veröffentlicht einen RSA-Modulus $n = pq$, wobei n ein Blum-Integer ist. Ein Prover P möchte einem Verifier V gegenüber beweisen, dass er in Besitz eines Geheimnisses ist. Dazu wählt P geheime $s \xleftarrow{\$} \mathbb{Z}_n$, $t_1 \xleftarrow{\$} \{-1, 1\}$ und veröffentlicht $v = t_1 \cdot s^2 \bmod n$.

Protokoll:

1. P wählt $r \xleftarrow{\$} \mathbb{Z}_n$, $t_2 \xleftarrow{\$} \{-1, 1\}$, berechnet $x = t_2 \cdot r^2 \bmod n$, sendet x an V.
2. V wählt $b \xleftarrow{\$} \{0, 1\}$, sendet b an P.
3. P berechnet $y = r \cdot s^b \bmod n$, sendet y an V.
4. V überprüft, ob $y^2 = \pm\, x \cdot v^b \bmod n$ gilt.

(i) Zeigen Sie die Korrektheit des Protokolls für ehrlichen P und V.

(ii) Falls V $b = 0$ wählt, ist die Antwort y von P unabhängig von dem Geheimnis s. Also kann ein Angreifer in diesem Fall auch ohne Kenntnis von s dafür sorgen, dass V akzeptiert indem er in (3.) als y einfach das selbst gewählte r sendet. Wie könnte ein Angreifer vorgehen, um V auch im Fall $b = 1$ zum Akzeptieren zu bringen. Wie hoch ist die Erfolgswahrscheinlichkeit bei k-maliger Wiederholung des Protokolls?

(iii) Zeigen Sie durch Angabe eines Simulators, dass die Zero-Knowledge-Eigenschaft gilt.

(iv) Welche Information würde V über v lernen, falls wir das Protokoll ohne die zufälligen Vorzeichen t_1, t_2 durchführen würden? (Überlegen Sie dazu, was P gegenüber V beweist, falls es keine Vorzeichen gibt).

Lösungsvorschlag zu Aufgabe 4.

(a) $\left(\frac{15}{35}\right) = \left(\frac{5}{35}\right)\left(\frac{3}{35}\right) = \left(\frac{5}{7}\right)\left(\frac{5}{5}\right)\left(\frac{3}{7}\right)\left(\frac{3}{5}\right) = 0$

$\left(\frac{32}{33}\right) = \left(\frac{-1}{33}\right) = (-1)^{\frac{33-1}{2}} = (-1)^{16} = 1$

$\left(\frac{17}{143}\right) = \left(\frac{17}{13}\right)\left(\frac{17}{11}\right) = \left(\frac{4}{13}\right)\left(\frac{6}{11}\right) = \left(\frac{2}{13}\right)\left(\frac{2}{13}\right)\left(\frac{2}{11}\right)\left(\frac{3}{11}\right) = 1 \cdot (2^5 \bmod 11)(3^5 \bmod 11) = 1 \cdot (-1) \cdot 1 = -1$

(b) $\left(\frac{-1}{n}\right) = \left(\frac{-1}{p}\right)\left(\frac{-1}{q}\right) = (-1)^{\frac{p-1}{2}}(-1)^{\frac{q-1}{2}} = (-1)(-1) = 1$, da $\frac{p-1}{2}$ bzw $\frac{q-1}{2}$ jeweils ungerade sind ($p \equiv q \equiv 3 \bmod 4$) und somit -1 kein quadratischer Rest bzgl p und q und somit auch nicht bzgl n ist.

(c) (i) Wenn P das Geheimnis s kennt, wird er V davon überzeugen können, da

$$y^2 = (rs^b)^2 = r^2 s^{2b} = t_2 x \cdot t_1 v^b = \pm x v^b \bmod n$$

(ii) Falls ein Angreifer $\mathcal{A}$ sich als P ausgeben möchte ohne s zu kennen, muss er V entsprechende x und y senden, so dass $y^2 = xv^b$. Da er die Wahl von b nicht kennt muss er im Vorfeld ein Bit b' raten um x passend aufzusetzen, indem er ein zufälliges y zieht und $x = y^2 v^{-b'} \bmod n$ berechnet. Dieses x schickt er an V. V zieht zufällig ein $b \xleftarrow{\$} \{0, 1\}$ und sendet es an $\mathcal{A}$. Daraufhin sendet $\mathcal{A}$ y an V, der überprüft ob $y^2 = xv^b = y^2 v^{-b'} v^b$ gilt. $\mathcal{A}$ ist erfolgreich, falls $b' = b$. Dies ist mit Wahrscheinlichkeit $\frac{1}{2}$ der Fall und somit hat $\mathcal{A}$ bei k Protokolldurchläufe eine Erfolgswahrscheinlichkeit von $(\frac{1}{2})^k$.

(iii) Ein Simulator S, der s nicht kennt, kann gültige Protokolltranskripte simulieren für alle möglichen V, indem er folgendes ausführt:

1. S wählt zufälliges Bit b', zufälliges $r \xleftarrow{\$} \mathbb{Z}_n$ und berechnet $x = r^2 v^{-b'} \bmod n$. Er sendet x an V.
2. V sendet zufälliges Bit b an S.
3. Falls $b' = b$ sendet S $y = r$ (siehe (ii)) ansonsten löscht er das Transkript und startet die Simulation von vorne.

(iv) Durch das Vorzeichen handelt es sich bei $v = t_1 s^2$ im Falle von $t_1 = 1$ um einen quadratischen Rest mit Quadratwurzel s oder für $t_1 = -1$ um keinen quadratischen Rest (siehe (b)). Würde man das Vorzeichen weglassen, würde die Information, dass v zur Menge der quadratischen Reste mod n gehört von P bewiesen werden. Das zweite Vorzeichen t_2 dient dazu, dies auch bei der Überprüfung von V zu verschleiern.