Karlsruher Institut für Technologie

Institut für Theoretische Informatik
Prof. Dr. D. Hofheinz

Stammvorlesung Sicherheit im Sommersemester 2017

Nachklausur

06.10.2017

Vorname: _____

Nachname: _____

Matrikelnummer: _____

Klausur-ID: _____

Hinweise

- Schreiben Sie auf **alle Blätter** der Klausur und **alle Zusatzblätter** Ihre Matrikelnummer und Ihren Namen.
- Für die Bearbeitung stehen Ihnen 60 Minuten zur Verfügung.
- Es sind keine Hilfsmittel zugelassen.
- Schreiben Sie Ihre Lösungen auf die Aufgabenblätter sowie auf deren Rückseiten.
- Bitte kennzeichnen Sie deutlich, welche Lösung gewertet werden soll. Bei mehreren angegebenen Möglichkeiten wird jeweils die schlechteste Alternative gewertet.
- Zusätzliches Papier erhalten Sie bei Bedarf von der Aufsicht.
- Die Klausur umfasst 12 Seiten.

Aufgabe	mögliche Punkte					erreichte Punkte				
	a	b	c	d	Σ	a	b	c	d	Σ
1	5	6	-	-	11			-	-	
2	4	3	1	2	10					
3	10	2	2	-	14				-	
4	1	3	2	5	11					
5	3	6	5	-	14				-	
Σ					60					

Aufgabe 1. (5 + 6 Punkte)

Wir betrachten das RSA-Verschlüsselungsverfahren aus der Vorlesung (ohne Padding).

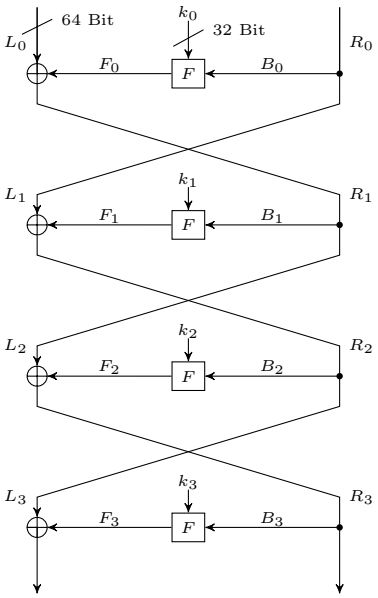
(a) Seien $P = 17$, $Q = 23$ und $N = P \cdot Q = 391$. Ferner sei der öffentliche Exponent $e = 61$ gewählt. Berechnen Sie den zugehörigen geheimen Exponenten d. Geben Sie alle Zwischenschritte an.

(b) Sei $N = P \cdot Q$ ein RSA-Modulus für ungerade, hinreichend große Primzahlen $P \neq Q$. Beweisen Sie folgende Aussage:

> Falls ein PPT-Algorithmus $\mathcal{A}$ existiert, der bei Eingabe N mit nicht-vernachlässigbarer Wahrscheinlichkeit $\epsilon_{\mathcal{A}}$ den Wert $\varphi(N)$ berechnet, dann existiert ein PPT-Algorithmus $\mathcal{B}$, der bei Eingabe N mit nicht-vernachlässigbarer Wahrscheinlichkeit $\epsilon_{\mathcal{B}}$ die Faktorisierung von N berechnet.

Geben Sie dazu insbesondere die Erfolgswahrscheinlichkeit ϵ_B an und begründen Sie, wieso diese nicht-vernachlässigbar ist. Argumentieren Sie zudem, wieso die Laufzeit des Algorithmus $\mathcal{B}$ polynomiell ist.

Aufgabe 2. $(4 + 3 + 1 + 2$ Punkte)
Gegeben sei folgendes Feistel-Netzwerk:



Die Länge der Nachrichten und Chiffrate beträgt jeweils 128 Bit und jeder Rundenschlüssel hat eine Länge von 32 Bit. Über die verwendete F-Funktion ist bekannt, dass die Parität des 4. Eingabebits zusammen mit den 16. und 42. Ausgabebits immer ungerade ist. Das heißt, dass für alle $k \in \{0,1\}^{32}$ und $b, f \in \{0,1\}^{64}$ mit $F(k,b) = f$ folgende Gleichung erfüllt ist:

$$b[4] \oplus f[16] \oplus f[42] = 1.$$

(a) Erweitern Sie diese Beobachtung auf einen Zusammenhang zwischen L_0, R_0, L_3 und R_3. Geben Sie Ihren Rechenweg an. Diesen Zusammenhang nennt man auch eine 3-Runden-Charakteristik.

Hinweis: Sie dürfen die Kurzschreibweise aus der Übung verwenden.

(b) Die 3-Runden-Charakteristik aus Aufgabenteil (a) ermöglicht es einen Angriff auf den Rundenschlüssel k_3 durchzuführen. Beschreiben Sie, wie dieser Angriff funktioniert.

Sie müssen das weitere Vorgehen um die Rundenschlüssel k_0, k_1, k_2 zu berechnen nicht beschreiben.

(c) Welche Informationen müssen einem Angreifer für den Angriff aus Aufgabenteil (b) zur Verfügung stehen?

(d) Funktioniert der Angriff aus Aufgabenteil (b) auch dann, wenn die 3-Runden-Charakteristik zusätzlich von mindestens einem Bit der Schlüssel k_0, k_1, k_2 abhängt? Die 3-Runden-Charakteristik gibt nun also einen Zusammenhang zwischen Bits aus L_0, R_0, L_3, R_3, k_0, k_1 und k_2 an. Begründen Sie Ihre Antwort.

Aufgabe 3. $(10(= 2 + 1 + 7) + 2 + 2$ Punkte)

(a) Betrachten wir nun kryptographische Hashfunktionen.

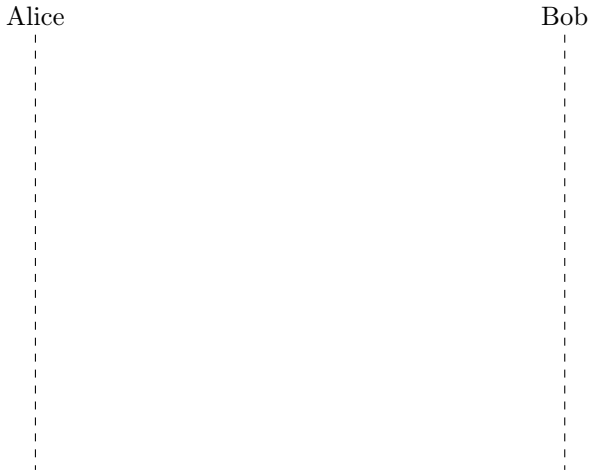
(i) Sei $H\colon \{0,1\}^* \to \{0,1\}^k$ eine über k parametrisierte effizient berechenbare Abbildung. Definieren Sie formal, wann H eine Einwegfunktion bezüglich der Urbildverteilungen $\{U_k\}_{k\in\mathbb{N}}$ ist.

(ii) Sei $H\colon \{0,1\}^* \to \{0,1\}^k$ eine kryptographische Hashfunktion. Welchen Aufwand hinsichtlich Rechenzeit und Speicherbedarf erfordert der Birthday-Angriff aus der Vorlesung gegen die Hashfunktion H?

(iii) Sei $H\colon \{0,1\}^* \to \{0,1\}^k$ eine kollisionsresistente Hashfunktion. Konstruieren Sie aus H eine kollisionsresistente Hashfunktion $\widetilde{H}\colon \{0,1\}^* \to \{0,1\}^{\widetilde{k}}$, die keine Einwegfunktion bezüglich der Urbildverteilungen $\{U_k\}_{k\in\mathbb{N}}$ ist, wobei mit U_k die Gleichverteilung auf $\{0,1\}^k$ gemeint ist. Die Ausgabelänge $\widetilde{k}$ von $\widetilde{H}$ soll dabei höchstens $2k$ sein.

Beweisen Sie, dass Ihre Konstruktion für $\widetilde{H}$ kollisionsresistent aber keine Einwegfunktion bezüglich der Urbildverteilungen $\{U_k\}_{k\in\mathbb{N}}$ ist.

(b) In der Vorlesung wurde das Diffie-Hellman-Schlüsselaustauschverfahren eingeführt. Dazu sei G eine Gruppe mit primer Ordnung p und Erzeuger g. Ergänzen Sie den Ablauf dieses Verfahrens zwischen den zwei Parteien Alice und Bob in der Zeichnung unten. Instantiieren Sie das Verfahren in der Gruppe G.



(c) Betrachten wir nun die in der Vorlesung vorgestellte Sicherheitslücke Cross-Site Scripting (XSS). Beschreiben Sie diese Sicherheitslücke im Allgemeinen.

Aufgabe 4. $(1 + 3 + 2 + 5$ Punkte)

Betrachten wir nun Verfahren zur symmetrischen Authentifikation von Nachrichten.

(a) Sei $H\colon \{0,1\}^* \to \{0,1\}^k$ eine beliebige kollisionsresistente Hashfunktion. Geben Sie an, wie beim HMAC-Verfahren aus der Vorlesung der Message Authentication Code für eine Nachricht $M \in \{0,1\}^*$ berechnet wird.

(b) Sei $F\colon \{0,1\}^{2k} \to \{0,1\}^k$ eine kollisionsresistente Kompressionsfunktion. Verwenden Sie die Merkle-Damgård-Konstruktion um aus F eine kollisionsresistente Hashfunktion H zu konstruieren. Definieren Sie H formal, eine Zeichnung ist nicht ausreichend.

(c) Sei H die Hashfunktion aus Aufgabenteil (b). Wir betrachten das MAC-Verfahren, bei dem der Message Authentication Code σ zu einer Nachricht $M \in \{0,1\}^*$ durch $\sigma := H(K \,\|\, M)$ berechnet wird, wobei $K \in \{0,1\}^k$ ein zufällig gewählter Bitstring ist, der nur dem Sender und dem Empfänger der Nachricht bekannt ist.

Ist dieses MAC-Verfahren EUF-CMA-sicher? Begründen Sie Ihre Antwort kurz.

(d) Sei $E\colon \{0,1\}^k \times \{0,1\}^k \to \{0,1\}^k$ eine Blockchiffre.

Wir betrachten folgendes MAC-Verfahren (Sig, Ver) für Nachrichten aus $\{0,1\}^*$, deren Länge ein Vielfaches von k ist:

$\mathsf{Sig}(K, M)$

$l := \frac{|M|}{k}$

$M =: M_1 \,\|\, M_2 \,\|\, \ldots \,\|\, M_l$ (wobei $M_i \in \{0,1\}^k$)

$\gamma_0 := 0^k$

$\gamma_i := E(K, M_i \oplus \gamma_{i-1})$ für $1 \leq i \leq l$

return $\sigma := \gamma_l$

$\mathsf{Ver}(K, M, \sigma)$

$l := \frac{|M|}{k}$

$M =: M_1 \,\|\, M_2 \,\|\, \ldots \,\|\, M_l$ (wobei $M_i \in \{0,1\}^k$)

$\gamma_0 := 0^k$

$\gamma_i := E(K, M_i \oplus \gamma_{i-1})$ für $1 \leq i \leq l$

if $\gamma_l == \sigma$ **then**

  return 1

else

  return 0

Beweisen Sie, dass dieses MAC-Verfahren nicht EUF-CMA-sicher ist, indem Sie einen EUF-CMA-Angreifer konstruieren. Zeigen Sie, dass Ihr EUF-CMA-Angreifer polynomielle Zeitkomplexität und eine nicht-vernachlässigbare Erfolgswahrscheinlichkeit im EUF-CMA-Spiel hat.

Aufgabe 5. (3 + 6 + 5 Punkte)

Sei $G = (V, E)$ ein ungerichteter Graph. Der Einfachheit halber sei $V = \{1, \ldots, n\}$ für $n \in \mathbb{N}$. Ein Hamiltonkreis in G ist ein Kreis in G, der jeden Knoten von G genau einmal besucht. (Ein Pfad kann hier als eine Folge von Knoten $(v_1, v_2, \ldots, v_l)$ aufgefasst werden, sodass für alle $1 \leq i \leq l-1$ die Kanten $\{v_i, v_{i+1}\}$ in E sind. Ein Kreis ist ein Pfad bei dem der Anfangs- und der Endknoten identisch sind.)

Wir betrachten ein Public-Key-Identifikationsschema (Gen, Prove, Ver), bei dem ein „Prover" Prove einen „Verifier" Ver davon überzeugen will, dass er einen Hamiltonkreis in einem vorgegebenen Graphen G kennt, ohne dabei Informationen über diesen Hamiltonkreis zu verraten.

- Die Parametergenerierung $\mathsf{Gen}(1^k)$ erzeugt einen Graphen $G = (V, E)$ zusammen mit einem zugehörigen Hamiltonkreis $C \in V^{|V|+1}$. Schließlich wird der öffentliche Schlüssel $pk := G$ und der geheime Schlüssel $sk := (G, C)$ ausgegeben. (Dieser Schritt wird von einer vertrauenswürdigen Instanz zu Beginn einmalig ausgeführt.)

Prove erhält als Eingabe den geheimen Schlüssel sk, Ver erhält als Eingabe den öffentlichen Schlüssel pk. Der Protokollablauf ist wie folgt definiert:

1.) Prove wählt eine zufällige Permutation $\varphi\colon V \to V$ der Knoten von G (also eine zufällige Umbenennung der Knoten von G), berechnet $H := \varphi(G)$ und committet sich auf die Kanten von H.[1] (Mit $\varphi(G)$ ist der Graph $\varphi(G) := (V, E')$ gemeint mit $E' := \{\{\varphi(u), \varphi(v)\} \mid \{u, v\} \in E\}$.)

2.) Ver wählt ein Bit $b \leftarrow \{0, 1\}$ zufällig gleichverteilt und sendet es an Prove.

3.) Ist $b = 0$, so sendet Prove den verwendeten Isomorphismus φ an Ver und öffnet die Commitments aus Schritt 1.).[2]
Ist $b = 1$, so sendet Prove den Hamiltonkreis $\varphi(C)$ in H an Ver und öffnet die Commitments für jene Kanten, die in $\varphi(C)$ verwendet werden.[3] (Für einen Pfad $C = (v_1, v_2, \ldots, v_l)$ ist mit $\varphi(C)$ der Pfad $(\varphi(v_1), \varphi(v_2), \ldots, \varphi(v_l))$ gemeint.)

4.) Ver akzeptiert genau dann, wenn die jeweiligen Commitments korrekt geöffnet wurden und eine der folgenden Bedingungen gilt:
   - Es gilt $b = 0$ und $\varphi(G) = H$.
   - Es gilt $b = 1$ und der Pfad $\varphi(C)$ ist ein Hamiltonkreis in H (um dies zu überprüfen, verwendet Ver die geöffneten Commitments auf die benötigten Kanten von H).

(a) Begründen Sie kurz, wieso das obige Protokoll korrekt ist.

[1] Sei Com ein Commitment-Verfahren. Prove committet sich auf die Kanten von H, indem er die Menge $\{\mathsf{Com}(e; R_e) \mid e \text{ Kante von } H,\ R_e \text{ zufällig}\}$ an Ver sendet.

[2] Prove öffnet die Commitments auf die Kanten von H, indem er $\{(e, R_e) \mid e \text{ Kante von } H\}$ an Ver sendet

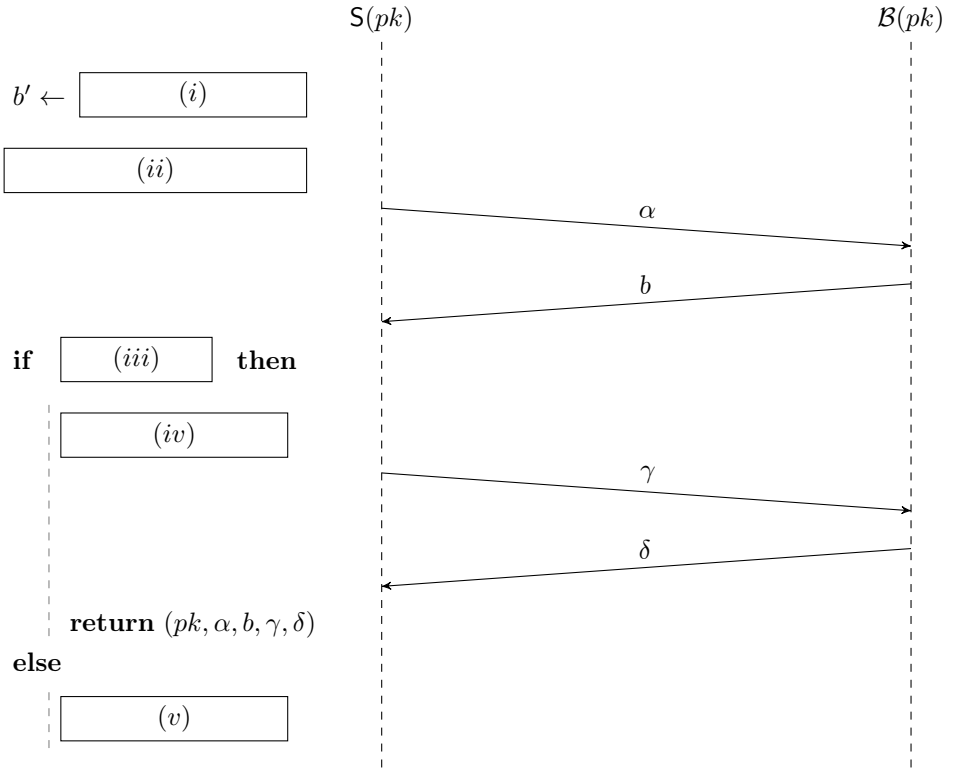
[3] Prove öffnet die Commitments auf die Kanten von $\varphi(C)$, indem er $\{(e, R_e) \mid e \text{ Kante von } \varphi(C)\}$ an Ver sendet.

(b) In diesem Aufgabenteil nehmen wir an, dass bereits im Voraus bekannt ist, welches Bit b der Verifier in Schritt 2.) wählen wird. Dann können wir für $b \in \{0,1\}$ jeweils einen PPT-Angreifer $\mathcal{A}_b$ in der Rolle des Provers konstruieren, der nur den öffentlichen Schlüssel $pk = G$ als Eingabe erhält und den Verifier trotzdem zum Akzeptieren bringt.

Geben Sie das Vorgehen von $\mathcal{A}_0$ und $\mathcal{A}_1$ in Schritt 1.) und Schritt 3.) an. Begründen Sie insbesondere, wieso Ihr $\mathcal{A}_b$ den Verifier Ver zum Akzeptieren bringt, wenn dieser in Schritt 2.) das Bit b ausgibt. (Sie müssen nicht begründen, wieso Ihr $\mathcal{A}_b$ polynomielle Laufzeit hat.)

Hinweis: Nehmen Sie dazu an, dass ein PPT-Algorithmus $\mathtt{GenHC}(n, m)$ gegeben ist, der einen zufälligen Graphen $\widetilde{G}$ mit n Knoten und m Kanten zusammen mit einem zugehörigen Hamiltonkreis $\widetilde{C}$ ausgibt.

(c) Sei nun $\mathcal{B}$ ein PPT-Angreifer in der Rolle des Verifiers. Geben Sie einen Simulator S an, der Transkripte ausgibt, die von Transkripten eines ehrlichen Provers (der sk kennt) mit $\mathcal{B}$ in der Rolle des Verifiers nicht unterscheidbar sind. Ergänzen Sie dazu die Beschriftungen in folgendem Diagramm, indem Sie die unten stehende Tabelle ausfüllen. Achten Sie darauf, dass die erwartete Laufzeit von S höchstens polynomiell ist.



(i)	
(ii)	
(iii)	
(iv)	
(v)	

Hinweis: Verwenden Sie Aufgabenteil (b). Sie dürfen Aufgabenteil (b) auch dann verwenden, wenn Sie dafür keine Lösung haben.