Aufgabenblatt 4
Sicherheit digitaler Systeme

Institut: Berliner Hochschule für Technik
Dozent: Prof. Dr. Christian Forler
Url: `https://lms.bht-berlin.de/`
Email: `cforler(at)bht-berlin.de`
Wintersemester 23/24

Aufgabe 1 Anwendungsgebiete
Nennen Sie drei Anwendungsgebiete der asymmetrischen Kryptographie

Aufgabe 2 Rechnen in Gruppen

a) Berechnen Sie $2^{10101} \bmod 11$ ohne Taschenrechner.

b) Berechnen Sie $3^{242} \bmod 35$ ohne Taschenrechner.

Aufgabe 3 Diffie-Hellman Protokolldurchlauf
Berechnen und skizzieren Sie einen DH-Protokolldurchlauf (d.h. A, B und g^{ab}) für die folgenden Parameter: $\mathbf{p} = 13, \mathbf{g} = 2, \mathbf{a} = 4, \mathbf{b} = 5$.

Aufgabe 4 Schlüsselmanagement

a) Wie viele **asymmetrische** Schlüssel werden benötigt wenn in einem Netzwerk, jeder Teilnehmer mit jeden anderen Teilnehmer (vollvermaschtes Netz) verschlüsselt kommunizieren möchte. Erläutern Sie ihre Antwort.

   (i) 10 Teilnehmer
   (ii) 100 Teilnehmer

b) Wie viele **symmetrische** Schlüssel werden benötigt wenn in einem Netzwerk, jeder Teilnehmer mit jeden anderen Teilnehmer (vollvermaschtes Netz) verschlüsselt kommunizieren möchte. Erläutern Sie ihre Antwort.

   (i) 10 Teilnehmer
   (ii) 100 Teilnehmer

Aufgabe 5 RSA-Protokoll

a) Entschlüsseln Sie den RSA-Chiffretext $C = 2$ für die folgenden RSA-Parameter $\mathbf{p} = 11, \mathbf{q} = 13, \mathbf{d} = 9$.

b) Verschlüsseln sie den Wert 142 für die folgenden Parameter $\mathbf{p} = 11, \mathbf{q} = 13, \mathbf{e} = 7$. Was fällt ihnen auf?

c) Was ist der Klartext von $C = 220$ für die öffentlichen RSA-Parameter ($\mathbf{e} = 7, \mathbf{n} = 221$). Erläutern Sie ihre Antwort.

Aufgabe 6 RSA-Exponent
Gegeben sind die RSA-Parameter $p = 11, q = 13$.

a) Ist $\mathbf{e} = 4$ eine gute Wahl für den öffentlichen RSA-Exponenten? Erläutern Sie ihre Antwort.

b) Welche Werte aus der Menge $S = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\}$ sind eine gute Wahl für $\mathbf{e}$. Erläutern Sie ihre Antwort.

Aufgabe 7 Digitale Signatur

a) Erläutern sie die Funktionsweise einer Digitalen Signatur

b) In wie fern unterscheiden sich eine digitale Signature von einem MAC.

Aufgabe 8 Hybride Verschlüsselung

a) Erläutern sie die Funktionsweise der hybriden Verschlüsselung.

b) Weshalb setzen die meisten kryptographischen Protokolle hybride Verschlüsselung ein?

Aufgabe 9 Programmieraufgabe: RSA-Faktorisierung (4 Punkte)
Die Faktorisierung eines RSA-Modulus n mittels $\varphi(n)$ ist einfach.

- berechne $z = n - \varphi(n) = n - (p-1)(q-1) = n - (pq - p - q + 1) = p + q - 1$
- berechne $x = (z+1)^2 = (p+q)^2$
- berechne $y = (z+1)^2 - 4n = (p-q)^2$
- berechne $p = (\sqrt{x} + \sqrt{y})/2$
- berechne $q = n/p$

Schreiben Sie ein Programm, welches als Kommandozeilenparameter einen RSA-Modulus n und $\varphi(n)$ übergeben bekommt. Das Programm soll die beiden Primfaktoren p und q berechnen und ausgeben.

```
Beispielaufruf: # factorize 51959 51504
233, 223
```

Aufgabe 10 Verschlüsselte Dateiübertragung (Teil 3) (8 Punkte)

Im folgenden sollen Sie ihre Lösung so anpassen, das Client und Server vor der Datenübertragung einen Sitzungsschlüssel aushandeln und dem die übertragende Datei verschlüsselt wird.
Verwenden Sie die kryptographischen Bibliothek Libsodium um den Schlüsselaustausch durchzuführen. Lesen Sie sich dazu die folgende Dokumentation durch `https://libsodium.gitbook.io/doc/key_exchange`.
Damit ist ihre Lösung sicher gegen passive Angreifer, aber immer noch anfällig für Man-In-the-Middle Angriffe.

Aufgabe 11 Verschlüsselte Dateiübertragung (Teil 4) (4 Punkte)
Schreiben Sie ein Programm `gen_signkey.c` welches mit Hilfe der kryptographischen Bibliothek Libsodium ein Signatur-Schlüsselpaar für den Server generiert. Lesen Sie sich dazu die folgende Dokumentation durch `https://libsodium.gitbook.io/doc/public-key_cryptography/public-key_signatures`. Ihr Programm soll den öffentlichen Signaturschlüssel in der Datei `server.pk` und den privaten Signaturschlüssel in der Datei `server.sk` ablegen.

Aufgabe 12 Verschlüsselte Dateiübertragung (Teil 5) (8 Punkte)
Der Server soll bei dem Schlüsselaustausch seinen öffentlichen Schlüssel pk mit seinem privaten Signaturschlüssel signieren und dem Client pk und Signatur s senden.
Der Client soll mit hilfe des öffentlichen Signaturschlüssels das Servertupel (pk, s) verifizieren.
Verwenden Sie die kryptographischen Bibliothek Libsodium um den Schlüsselaustausch durchzuführen. Lesen Sie sich dazu die folgende Dokumentation durch `https://libsodium.gitbook.io/doc/public-key_cryptography/public-key_signatures`.
Der Client soll die Übertragung abbrechen, falls die Verifikation nicht erfolgreich war, da dies ein starkes Indiz für einen MitM-Angriff ist.
Nun haben Sie ein Client-Server-Anwendung geschrieben mit der Sie sicher Dateien über ein unsicheres Transportnetzt (z.B. dem Internet) übertragen können.