## Aufgabe 1: Zentrale Begriffe der Kryptographie (5 Punkte)

**1.1 Unterschiedliche Chiffren (Pflicht; 2 Punkte)** – Was ist der Unterschied zwischen einem symmetrischen und einem asymmetrischen Kryptosystem?

Bei einem symmetrischen Kryptosystem ist der Schlüssel der zum Verschlüsseln der Nachricht verwendet wird, derselbe wie der Schlüssel der zum Entschlüsseln verwendet wird. Bei einem asymmetrischen Kryptosystem werden zum Ver- und Entschlüsseln zwei verschiedene Schlüssel verwendet (Private und Public Key).

Symmetrisch: $E_{Key}\left(V_{Key}(N)\right)$

Asymmetrisch: $E_{secret}\left(V_{Public}(N)\right)$

**1.2 Hybride Kryptosysteme (Pflicht; 3 Punkte)** – Alice (A) und Bob (B) haben jeweils ein Schlüsselpaar erzeugt, um vertraulich miteinander kommunizieren zu können. Sie verfügen nun jeweils über einen öffentlichen (public) Schlüssel ($K_p^A$ bzw. $K_p^B$) sowie einen privaten (secret) Schlüssel ($K_s^A$ bzw. $K_s^B$). Ihre öffentlichen Schlüssel haben die beiden bereits bei einem persönlichen Treffen ausgetauscht. Alice möchte Bob nun eine vertrauliche Nachricht übermitteln.

**a) Unter welchen Umständen wird Alice dazu auf ein sog. hybrides Kryptosystem zurückgreifen?**

Ein Hybrides Kryptosystem profitiert von dem einfachen Schlüsselaustausch bei Asymmetrischer Verschlüsselung und der hohen Verschlüsselungsleistung der symmetrischen Verschlüsselung. Wenn also viele Daten verschickt werden müssen, macht es Sinn auf ein Hybrides Kryptosystem zurückzugreifen.

**b) Wie geht Alice im Detail vor, wenn sie ein hybrides Kryptosystem einsetzt?**

Das Asymmetrische Kryptosystem wird zum Austausch eines symmetrischen (Sitzungs-) Schlüssels verwendet. Die eigentliche Nachricht wird dann mit dem Sitzungsschlüssel verschlüsselt. Das kann z.B. so ablaufen: Alice generiert einen Sitzungsschlüssel $k$, den sie mit dem öffentlichen Schlüssel von Bob verschlüsselt und an diesen sendet. Bob kann mit seinem privaten Schlüssel die Nachricht entschlüsseln und kennt somit auch den Sitzungsschlüssel $k$. (An dieser Stelle kann Bob den Erhalt von $k$ bestätigen, indem er $k$ nochmal mit dem öffentlichen Schlüssel von Alice verschlüsselt und an sie schickt) Jetzt kann Alice eine Nachricht mit $k$ verschlüsseln und an Bob schicken. Er kennt ebenfalls $k$ und kann die Nachricht entschlüsseln.

**c) Wie sieht die übertragene Nachricht in diesem Fall aus?**

Die Nachricht wurde mit $k$ symmetrisch verschlüsselt. Also $V_k(N)$.

## Aufgabe 2: Parkhaus (8 Punkte)

In dem kostenpflichtigen Parkhaus eines Einkaufszentrums einer bayrischen Großstadt kommt das neue iPark secure zum Erheben der Parkgebühren zum Einsatz. Dieses System gibt beim Einfahren in das Parkhaus an einer Schranke Parktickets auf Karton aus. Auf die Tickets werden Barcodes aufgebracht. Vor dem Verlassen des Parkhauses muss an einem Kassenautomat die Parkgebühr entrichtet werden. Die genaue Höhe wird mit Hilfe der ausgegebenen Karte ermittelt. Die Schranke bei der Ausfahrt öffnet nur, wenn zwischen dem Bezahlvorgang weniger als 10 Minuten verstrichen sind.

Neben einer Vielzahl von Läden gibt es in dem Einkaufszentrum jedoch zwei Unternehmen, welche ihren Kunden besondere Park-Konditionen einräumen möchten: Ein Einzelhändler bietet an, ab einem Einkaufswert von 15 EUR das Parkticket mit einer Markierung zu versehen (dazu später mehr), die den Kunden 90 Minuten kostenfreies Parken ermöglicht. Ein angeschlossenes Kino erlaubt es hingegen seinen Besuchern (ebenfalls durch Markierung des Parktickets), ihre Fahrzeuge für pauschal 2,50 EUR den ganzen Tag im Parkhaus unterzustellen.

Sie haben sich acht bereits verwendete Parktickets besorgt (siehe Abb. 1). Sie wissen, dass der letzte und vorletzte Barcode (also die beiden rechten Barcodes) bereits durch die Schranke an der Einfahrt auf das Ticket aufgedruckt werden. Der dritte Code von rechts wird nach dem Bezahlvorgang durch den Kassenautomat hinzugefügt – ebenso wie die menschenlesbaren Informationen in der Ecke links unten. Hat man das Kino oder das Geschäft mit den vergünstigten Parkkonditionen besucht, so wird dort noch jeweils ganz links ein Barcode aufgebracht (dieser ist nicht auf allen Tickets vorhanden).

**1. Funktionsweise (Optional)** – Untersuchen und vergleichen Sie die Parktickets und versuchen Sie zu verstehen, wie das System arbeitet. Wie funktioniert es, welche Daten werden vermutlich wie übermittelt und warum wurde es auf diese Art ausgestaltet?

…

**2. Sicherheitsanalyse (Pflicht; 4 Punkte)** – Weist das System Schwächen auf? Welches Angreifermodell liegt dem System demnach offenbar zu Grunde?

Ja, das System weist Schwächen auf. Eine offensichtliche ist, dass man ohne das entsprechende Geschäft besucht zu haben, den Barcode für eine Vergünstigung selber aufdrucken kann, da er für die einzelnen Geschäfte immer gleich zu sein scheint. Bei genauerem Hinschauen fällt sogar auf, dass der Barcode, der ein Ticket als „Bezahlt“ markiert immer der gleiche ist. Somit kann man sich nicht nur eine Vergünstigung aufdrucken, sondern direkt den Parkschein als bezahlt markieren, in dem man den entsprechenden dritten Barcode aufdruckt. Das Angreifermodell sieht wie folgt aus:

- Rolle des Angreifers: Benutzer des Systems
- Verbreitung des Angreifers: Alle Eingriffe sind mit Hilfe des Parktickets realisierbar.
- Verhalten des Angreifers: Aktiv, Verändernd. Das Parkticket, das man bei der Einfahrt erhält, kann man so manipulieren, dass man vergünstigt/kostenlos parken kann (selber bedrucken)
- Rechenkapazität: sehr gering, Angreifer muss das Muster erkennen (hat bei mir keine 60 Sekunden gedauert)

Alle diese Ausführungen beruhen auf der Annahme, dass die Informationen zwischen Bezahlsystem/Geschäft/Schranke nur über das Ticket/Barcode ausgetauscht werden. Wenn diese Geräte in einem Netz miteinander verbunden wären, würde das System deutlich sicherere werden.

**3. Umsetzung mit kryptographischen Techniken (Pflicht; 4 Punkte)** – Nehmen Sie nun an, dass es nicht möglich ist, Daten zwischen den Komponenten des Systems auf einem anderen Weg als auf dem Ticket zu übermitteln. Wie würden Sie das System durch Einsatz von kryptographischen Techniken gestalten, um Betrug in diesem Fall effektiv zu verhindern? Beschreiben Sie Ihr Parksystem im Detail, auch gerne unter Zuhilfenahme einer Abbildung, und begründen Sie Ihre Entscheidungen.

Eine einfache Erweiterung des Systems wäre alle Daten, die mit dem Barcode dargestellt werden, mit einem symmetrischen Kryptosystem zu verschlüsseln (alle Komponenten können diesen Schlüssel dann verwenden). Des Weiteren dürfen die Markierungen für Bezahlt und Vergünstigungen nicht immer die gleichen sein. Es müssten zufällige Daten angehängt werden.

Eine andere Möglichkeit wäre, beim Bezahlen des Tickets eine Checksumme über alle Daten zu bilden und diese, symmetrisch verschlüsselt, als „Bezahlungscode“ aufzudrucken. Die Schranke könnte dann prüfen, ob die Checksumme gültig ist.

# Aufgabe 3: Authentifizierungsprotokolle (4 Punkte)

**1. Verschlüsselte Passwort-Übermittlung (Optional)**

Bei einem passiven Angriff könnte der in der Vergangenheit festgelegte Schlüssel (der in Klartext übertragen werden muss) ausgelesen werden. Ein passiver Angreifer kann auch den verschlüsselten Users und Passworts c abfangen, da der sich beim wiederholten Anmelden nicht ändert (schwerwiegendes Problem). Auch das Knacken des wirklichen Passworts über Rainbow-Tables ist möglich.

Ein aktiver Angreifer kann per Brute-Force alle Möglichkeiten oder auch nur die am häufigsten vorkommenden Passwörter ausprobieren zu einem bestimmten User.

**2. Authentifikationssystem auf Basis indeterministischer symmetrischer Verschlüsselung (Pflicht; 2 Punkte)**

Die Ergänzung einer vom Client gewählten Zusatzzahl verbesserter die Sicherheit nicht. Ein passiver Angreifer kann weiterhin die Anmeldung auslesen und genau diese Kombination aus c mit der Zufallszahl r verwenden. Die Verwendung von Rainbow-Tables wird weniger attraktiv, da mit jedem neuen r eine neue Tabelle berechnet werden müsste.

Auch einen aktiven Angreifer schreckt dies nicht ab. Unabhängig von der Verwendung einer Zufallszahl bleibt diese Brute-Force-Angriffsform bestehen.

**3. Challenge-Response-Authentifizierung (Pflicht; 2 Punkte)**

Einen Brute-Force-Angriff kann dies immer noch nicht verteidigen, jedoch kann ein Angreifer jetzt nur noch eine einmal eingefangene Anmeldung benutzen, wenn nach der gleichen Zufallszahl gefragt wird.

**4. Sichere Challenge-Response-Authentifizierung (Optional)**

Wenn ein symmetrischer Schlüssel verwendet wird, dann die Challenge-Response-Authetifizierung zweimal durchgeführt werden. Einmal spielt der muss sich der Dienstleiter versichern, dass der Client ‚echt' ist (siehe Teilaufgabe 3), indem der Dienstleister die Zufallszahl erzeugt und das Ergebnis überprüft. Und ein zweiter Durchlauf, wo der Client sich von der Echtheit versichert und selbst eine Zufallszahl verschickt und das Ergebnis validiert.

Die Sicherheit basiert darauf, dass beide nur mit Kenntnis des Schlüssels die Authentifizierung c mit Hilfe der Zufallszahl r berechnen können, vorausgesetzt, die beiden r unterscheiden sich.

# Aufgabe 4: „Mensch ärgere Dich nicht" über das Telefon (optional)

# Aufgabe 5: RSA-Verfahren (6 Punkte)

**1. Grundlagen (Optional)**

Zur Verwendung von RSA sind zwei (möglichst große) unterschiedliche Primzahlen (p und q) nötig. Der öffentliche Schlüssel besteht aus dem Produkt der beiden Primzahlen und einer frei gewählten Zahl, die Teilerfremd zu (p-1)(q-1) ist (genannt e). Der private Schlüssel ist die multiplikative Inverse Zahl zur eben gewählten Zahl (genannt d). Verschlüsseln: c(m) = m^e mod p*q. Entschlüsseln: m(c)=m^e mod p*q.

Die Sicherheit besteht darin das es unendlich schwer ist, aus dem Produkt zweier Primzahlen die einzelnen Primzahlen zu berechnen. Durch die Primzahlen ist eine Primärfaktor-Zerlegung nicht möglich.

Verwendung bei der Verschlüsselung von Dokumenten und E-Mail, Public-Private-Key-Verfahren und vieles mehr.

**2. Anwendung (Pflicht; 6 Punkte)**

n=p*q=281*289=109309

r=(p-1)*(q-1)=208*288=108640

e=67

d=3243 (Erweiterter Euklidischer Algorithmus mit n=67, p=108640)

e*d mod r = 67 * 3243 mod 108640 = 217281 mod 108640 = 1

**Code (HTML+JS):**

```
<html>
<body>
	<div id="result">Calculating…</div>
	<script src="https://github.com/silentmatt/javascript-
biginteger/raw/master/biginteger.js"></script>

	<script>
	var data = [103625, 71396, 5872, 102989, 10232, 36843, 71765, 5872,
10232, 14809, 108822, 108822, 69296, 32156, 36704, 105697, 71396, 25948,
71396, 102989, 10232, 25948, 71765, 64024, 36843, 10232, 16718, 105867,
36704, 34992, 5872, 64024, 36843, 5872, 10232, 2762, 73111, 5872, 19729,
5872, 64024, 10232, 109169, 71765, 1086, 73111, 57424, 71765, 34992, 60372,
10232, 108822, 1086, 73111, 71396, 57424, 40412, 40412, 71765, 5872, 36704,
5872, 82037, 10232, 86175, 64024, 34992, 102989, 5872, 71765, 16718, 5872,
102989, 19729, 105867, 36843, 5872, 36704, 36704, 5872, 82037, 10232,
61644, 105697, 71765, 64024, 36265, 105867, 109169, 10232, 2762, 105697,
36265, 36704, 5872, 25948, 82037, 10232, 36843, 71765, 5872, 10232, 89982,
27255, 64024, 69296, 62098, 108822, 71765, 1086, 73111, 5872, 102989,
73111, 5872, 71765, 57424, 10232, 61865, 105867, 64024, 10232, 35203,
105697, 25948, 25948, 109169, 105867, 5872, 102989, 57424, 5872, 102989,
64024, 10232, 71396, 64024, 36843, 10232, 36843, 105697, 40412, 71396,
34992, 5872, 73111, 105867, 5872, 102989, 71765, 34992, 5872, 10232, 86175,
64024, 34992, 102989, 71765, 16718, 16718, 5872, 82037, 10232, 57837,
71396, 34992, 105697, 64024, 34992, 25948, 69296, 10232, 71396, 64024,
36843, 10232, 57837, 71396, 34992, 102989, 71765, 16718, 16718, 25948,
78325, 105867, 64024, 57424, 102989, 105867, 36704, 36704, 5872, 82037,
10232, 102020, 71765, 105867, 19729, 5872, 57424, 102989, 71765, 25948,
1086, 73111, 5872, 10232, 52356, 5872, 102989, 16718, 105697, 73111,
102989, 5872, 64024, 82037, 10232, 2762, 71765,19729, 71765, 64024, 34992,
69296, 86175, 57424, 57424, 105697, 1086, 78325, 10232, 71396, 64024,
36843, 10232, 35203, 105867, 109169, 5872, 102989, 69296, 86175, 64024,
105697, 36704, 40103, 25948, 71765, 25948, 82037, 10232, 14809, 102989,
71396, 64024, 36843, 36704, 105697, 34992, 5872, 64024, 10232, 36843, 5872,
102989, 10232, 32156, 102989, 40103, 108306, 57424, 105867, 34992, 102989,
105697, 108306, 73111, 71765, 5872, 82037, 10232, 86175, 71396, 57424,
73111, 5872, 64024, 57424, 71765, 16718, 71765, 78325, 105697, 57424,
71765, 105867, 64024, 25948, 108306, 102989, 105867, 57424, 105867, 78325,
105867, 36704, 36704, 5872, 82037, 10232, 36843, 105697, 25948, 10232,
61644, 108822, 86175, 69296, 52356, 5872, 102989, 16718, 105697, 73111,
```

```
102989, 5872, 64024, 10232, 71396, 64024, 36843, 10232, 64024, 105697,
57424, 71396, 5872, 102989, 36704, 71765, 1086, 73111, 10232, 105697,
36704, 36704, 5872, 10232, 105697, 64024, 36843, 5872, 102989, 5872, 64024,
10232, 59390, 64024, 73111, 105697, 36704, 57424, 5872, 82037, 10232,
36843, 71765, 5872, 10232, 109169, 71765, 102989, 10232, 71765, 64024,
10232, 36843, 5872, 102989, 10232, 27255, 5872, 36265, 71396, 64024, 34992,
10232, 71396, 64024, 36843, 10232, 36843, 5872, 102989, 10232, 52356,
105867, 102989, 36704, 5872, 25948, 71396, 64024, 34992, 10232, 36265,
5872, 73111, 105697, 64024, 36843, 5872, 36704, 57424, 10232, 73111,
105697, 36265, 5872, 64024, 10232, 60372, 69296, 62098];

      var rsa_p=281;
      var rsa_q=389;
      var rsa_e=67;
      var rsa_d=3243;
      var rsa_N=rsa_p*rsa_q;
      var rsa_r=(rsa_p-1)*(rsa_q-1);

      var decoded = "";
      var rsa_e_big = BigInteger(rsa_e);
      var rsa_N_big = BigInteger(rsa_N);
      for(var i=0; i<data.length; i++) {
            var num = BigInteger(data[i]);
            var num_mod = num.modPow(rsa_e_big, rsa_N_big);
            var result = num_mod;
            decoded += result + ", ";
      }

      document.getElementById("result").innerHTML = decoded;
</script>
</body>
</html>
```

**3. Sichere Implementierung (Optional)**