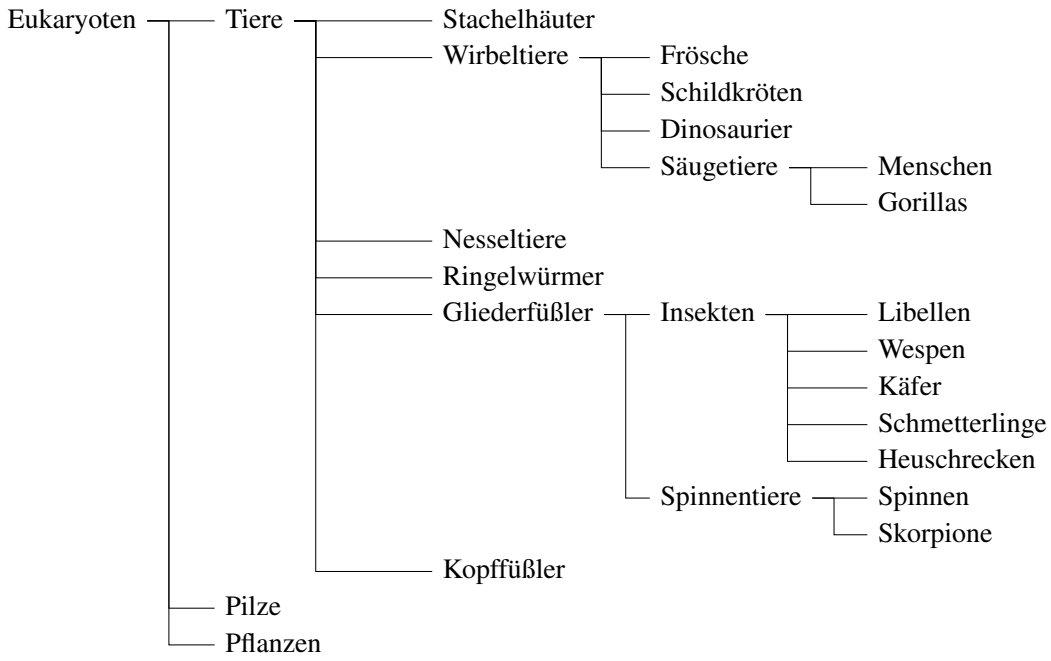
Bioinformatik: Konstruktion phylogenetischer Bäume

Gruppenaufgabe Tag 1

Ein *phylogenetischer Baum* (im Folgenden einfach nur *Baum* genannt) beschreibt, wie verschiedene Spezies sich in der Evolution entwickelt haben. Die Wurzel des Baumes bildet eine (in der Regel schon lange ausgestorbene) Urspezies. Durch Mutationen bildeten sich aus ihr neue Spezies aus, die die Kinder der Wurzel des Baumes darstellen – und in der Regel auch schon lange ausgestorben sind. Danach kam es durch immer neue Mutationen dazu, dass immer neue Spezies entstanden sind, bis wir schließlich die heutigen Spezies vorfinden:

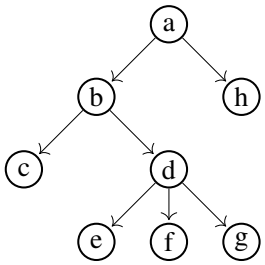


Leider ist es nicht so einfach, einen solchen Baum zu bestimmen: Dass die Schmetterlinge nicht vom Menschen abstammen, scheint irgendwie klar; dass aber Heuschrecken mehr mit Libellen gemeinsam haben als mit Spinnen, sieht man ihnen nicht direkt an. Aber auch wenn die direkte Abstammungslinie zwischen zwei Spezies schwer zu bestimmen ist, können Biologinnen und Biologen häufig so genannte *Triplets* bestimmen: Sie geben für drei Spezies an, dass zwei von ihnen enger miteinander verwandt sind als jeweils mit der dritten. Beispielsweise bilden Menschen (m), Gorillas (g) und Skorpione (s) ein Triplet, in dem Menschen und Gorillas untereinander enger verwandt sind als jeweils mit den Skorpionen – man schreibt hierfür $mg|s$. Ähnlich bilden Skorpione, Wespen und Frösche ein Triplet, bei dem Skorpione und Wespen enger zusammengehören, also $f|sw$. Damit ergibt sich als Ziel der *tripletbasierten Rekonstruktion von Bäumen,* zu einer Menge von Triplets einen Baum zu bestimmen, der zu allen von Biologen bestimmten Triplets passt.

Bevor Sie sich in die Algorithmen und Datenstrukturen stürzen, sollen das Modell und das Problem etwas formaler gefasst werden.

Ein *Baum* bezeichnet immer einen gerichteten Graphen, in dem es von einem bestimmten Knoten, die *Wurzel* genannt, genau einen Pfad zu jedem anderen Knoten gibt. Wir schreiben $u \to v$, wenn es eine Kante von u nach v gibt. Gibt es von einem Knoten u zu einem Knoten v einen Pfad (zur Not auch ohne Kanten, falls $u = v$), so heißt v ein *Nachfahre* von u (und u *Vorfahr* von v) und wir schreiben $u \to^* v$. Ist $u \neq v$, so schreiben wir $u \to^+ v$ und nennen v einen *echten* Nachfahren von u. Knoten ohne echte Nachfahren heißen *Blätter*. Für zwei Knoten u und v bilden die Knoten, die sowohl Vorfahren von u wie auch von v sind, einen Pfad. Den letzten Knoten auf diesem Pfad nennt man ihren *nächsten gemeinsamen Vorfahren* $\mathrm{lca}(u,v)$, wobei »lca« für *lowest common ancestor* steht.

Beispielsweise gelten im folgenden Baum $a \to b$ und $b \to^* b$ und $a \to^+ c$ und $b \to^* f$, aber weder $a \to c$ noch $b \to^+ b$ noch $h \to^* d$. Die Blätter sind gerade die Knoten c, e, f, g und h. Es gelten $\mathrm{lca}(c,f) = b$, $\mathrm{lca}(e,b) = b$, $\mathrm{lca}(h,g) = a$ und $\mathrm{lca}(a,a) = a$.



Ein Baum B *enthält das Triplet* $uv|w$, wenn u, v und w drei unterschiedliche Blätter sind und $\mathrm{lca}(u,w) = \mathrm{lca}(v,w) \to^+ \mathrm{lca}(u,v)$ gilt. Beispielsweise enthält der Baum des Lebens für $m =$ Menschen, $g =$ Gorillas und $w =$ Wespen das Triplet $mg|w$, da Tiere $= \mathrm{lca}(m,w) = \mathrm{lca}(g,w) \to^+ \mathrm{lca}(m,g) =$ Säugetiere gilt. Statt $mg|w$ schreiben wir auch $w|mg$ oder auch $gm|w$ oder $w|gm$, denn auf die Reihenfolge der Blätter kommt es nicht an. Als weitere Beispiele enthält der obige Baum die Triplets $c|eg$ und $cg|h$ und $bg|h$, hingegen nicht $ab|c$ und auch nicht $cf|g$ und auch nicht $ef|g$.

Aufgabe 0: Auftaktaufgabe

1. Finden Sie einen Baum B, der die Triplets $a|bc$, $bc|d$, $bc|e$, $ac|d$ enthält.
2. Zeigen Sie, dass es keinen Baum gibt, der sowohl das Triplet $ab|c$ wie auch das Triplet $bc|a$ enthält.
3. Finden Sie einen Baum B', der die Triplets $a|bc$, $bc|d$, $bc|e$, $ac|d$ enthält, das Triplet $ae|d$ hingegen nicht.

Die nachfolgenden Aufgaben 1 und 2 sowie 3 und 4 bauen jeweils aufeinander auf und sollten daher nacheinander angegangen werden. Sie sollten mit Aufgabe 1 anfangen, Sie können aber mit Aufgabe 3 beginnen, ohne Aufgaben 1 und 2 vollständig gelöst zu haben.

Aufgabe 1: Nachfahren schnell bestimmen

Algorithmen für (phylogenetische) Bäume müssen häufig für einen festen Baum B sehr oft die Frage beantworten »Ist u Vorfahre von v?« für unterschiedliche Knoten u und v. Ein Algorithmus kann diese Fragen sehr schnell beantworten, wenn er zunächst für den Baum die *Vorfahrenmatrix* berechnet: Dies ist einfach ein zweidimensionales Array, in dem für jedes Paar (u,v) vermerkt ist, ob u Vorfahre von v ist oder nicht. Hat man diese Datenstruktur einmal berechnet, kann die Frage »Gilt $u \to^* v$?« bei Eingabe (u,v) offenbar in konstanter Zeit, also Zeit $O(1)$, beantwortet werden.

Leider ist die Vorfahrenmatrix keine besonders effiziente Datenstruktur: Sie braucht Platz $O(n^2)$ bei n Blättern, und es braucht auch (mindestens) Zeit $O(n^2)$, sie zu erstellen. Sie sollen daher eine bessere Datenstruktur finden, die mit deutlich weniger als quadratischem Platz auskommt.

Entwickeln Sie daher folgende zwei Algorithmen: Der Algorithmus VORVERARBEITUNG bekommt einen Baum B als Eingabe und gibt eine Datenstruktur D aus, die möglichst klein sein soll (jedenfalls kleiner als quadratisch). Der Algorithmus BEANTWORTEFRAGE bekommt dann als Eingabe (B,D,u,v), wobei u und v Knoten im Baum B sind und D das Ergebnis Ihrer Vorverarbeitung für B ist. Er soll dann möglichst schnell die Frage »Gilt $u \to^* v$?« beantworten, wobei er natürlich die in D vorhandenen Informationen nutzen kann.

Geben Sie sowohl die Algorithmen wie auch den Aufbau von D an und schätzen Sie die Größe von D und die Laufzeiten der Algorithmen ab.

Aufgabe 2: Nächsten gemeinsamen Vorfahren schnell bestimmen

Diese Aufgabe ist fast identisch zur vorherigen, jedoch soll der Algorithmus BEANTWORTEFRAGE bei Eingabe (B,D,u,v) nun möglichst schnell $\operatorname{lca}(u,v)$ ausgeben.

(Diese Aufgabe ist deutlich schwieriger als die vorherige. Lösen Sie zunächst Aufgabe 1, bevor Sie sich an Aufgabe 2 versuchen.)

Aufgabe 3: Baum aus Triplets rekonstruieren

Entwickeln Sie einen möglichst schnellen Algorithmus, der das *Rekonstruktionsproblem* für Triplets löst. Die Eingabe für den Algorithmus ist eine Menge M von Triplets. Ihr Algorithmus soll einen Baum B bestimmen, der alle Triplets in M enthält, oder er soll feststellen, dass kein solcher Baum existiert.

(Der Algorithmus muss also insbesondere die ersten beiden Punkte der Auftaktaufgabe lösen können.)

Aufgabe 4: Baum für gewünschte und verbotene Triplets rekonstruieren

Diese Aufgabe ist fast identisch zur vorherigen, jedoch soll der Algorithmus als Eingabe zusätzlich zur Menge M der Triplets, die in B enthalten sein müssen, noch eine zweite Menge V an verbotenen Triplets bekommen, die in B *nicht* vorkommen dürfen.

(Der Algorithmus muss also insbesondere alle drei Punkte der Auftaktaufgabe lösen können).

IT-Sicherheit: Sichere Autoschlüssel

Gruppenaufgabe Tag 2

Kürzlich haben Sie angefangen, bei einer IT-Sicherheitsfirma zu arbeiten, die sich auf die Beratung von Automobilfirmen spezialisiert hat – und die Automobilfirmen haben Beratung auch nötig, denn moderne Autos sind im Wesentlichen fahrbare Computernetzwerke. Ein Beispiel der vielen elektronischen Komponenten eines heutigen Autos ist der Autoschlüssel: Man drückt auf einen Knopf am Schlüssel und das Auto geht auf.

Ihre Chefin gibt Ihnen als ersten Job zu analysieren, wie sicher die Kommunikation zwischen Autos und Autoschlüsseln bei verschiedenen Firmen ist. Später sollen Sie dann auch Vorschläge zur Verbesserung machen. Ihre Chefin gibt Ihnen allerdings noch folgende Hinweise mit auf den Weg:

1. Den Besitzer des Autos wollen wir Bob nennen. Nur Bobs Schlüssel soll das Auto öffnen können. Wir betrachten nur den Fall des Öffnens auf Knopfdruck am Schlüssel; Schließen, Starten des Motors und was ein Schlüssel noch alles kann sollen nicht behandelt werden.

2. Auto und Autoschlüssel kommunizieren immer über eine einfache Funkverbindung. Eine mit einem Laptop ausgerüstete Angreiferin, im Folgenden immer Eve genannt, kann diese Funkverbindung leicht und unauffällig abhören, stören oder selber nutzen – allerdings nur, wenn sie in der Nähe ist. Eve hat sich auch eine genaue Spezifikation der Hardware und der Software beschaffen können, die von der jeweiligen Firma verbaut wird; Eve kennt also die im Folgenden vorgestellten Protokolle genau.

3. Sowohl das Auto wie auch der Autoschlüssel können Berechnungen durchführen. Beispielsweise können beide leicht das bitweise Exklusiv-Oder (geschrieben $\oplus$) zweier Bitstrings ausrechnen; so ist $001101 \oplus 111000 = 110101$ und $0011 \oplus 0101 = 0110$. Ebenso können sie Zähler verwalten, nicht zu lange Schleifen ausführen oder mittels der Methode ALSBITSTRING beliebig viele Parameter wie Zahlen oder Texte in einen langen Bitstring verwandeln.

4. Das Auto hat deutlich mehr Rechenkraft (etwa wie ein normaler PC) als der Autoschlüssel. Sind komplexe Berechnungen auf dem Schlüssel nötig, so muss die Firma Spezialhardware bauen, und die sollte eher einfach und stromsparend sein. Insbesondere bevorzugen es die Firmen, wenn die Methoden ERZEUGEZUFALLSZAHL und CRYPTOHASH weiter unten *nicht* auf den Schlüsseln ausgeführt werden müssen; es sei denn, es geht nicht anders. Auf einem Autoschlüssel lassen sich kleine Datenmengen beliebig lesen und speichern; größere Datenmengen (über ein Kilobyte bis zu maximal einigen Megabyte) können nur als read-only Daten auf dem Autoschlüssel gespeichert sein.

5. Die Methode ERZEUGEZUFALLSZAHL erzeugt eine zufällige 256-Bit-Zahl. Man kann bei der Implementierung von Zufallszahlengeneratoren beliebig viel falsch machen, Sie *müssen* aber davon ausgehen, dass hier die Implementierung perfekt ist und die erzeugten Zahlen vollkommen zufällig sind. Einen Angriff auf die Interna der Methode *dürfen Sie nicht* betrachten.

6. Die Methode CRYPTOHASH nimmt eine beliebig lange Bitfolge x und liefert eine 256-Bit-Zahl z. Die Berechnung der Methode ist *nicht umkehrbar*, das heißt, es ist praktisch unmöglich, bei Eingabe z irgendein x zu errechnen, so dass CRYPTOHASH$(x) = z$ gilt. Wieder kann man bei der Implementierung von Hashfunktionen beliebig viel falsch machen, wieder *müssen* Sie aber annehmen, dass CRYPTOHASH perfekt implementiert ist und keinerlei Angriffe zulässt. Allerdings *ist* Eve die Funktion bekannt und auch sie kann sie berechnen, falls nötig – sie kann sie nur nicht umkehren.

7. Weitere, fortgeschrittenere Verfahren wie symmetrische oder asymmetrische Verschlüsselung sind den Firmen zu aufwendig. Sie *dürfen* im Folgenden *keine* Vorschläge machen, bei denen solche Verfahren direkt oder indirekt benötigt werden.

Die erste Firma, die Sie beraten sollen, heißt MG. Deren Chefentwickler hat sich Folgendes gedacht: »Drückt Bob auf den Schlüssel, lasse ich den Schlüssel einfach den Befehl ›Öffnen‹ zusammen mit der Seriennummer des Autos übertragen. Empfängt das Auto diesen Befehl zusammen mit der eigenen Serienummer, so geht es auf. Kann ich garantieren, dass unsere Seriennummern eindeutig sind, kann es nicht vorkommen, dass ein anderer als Bobs Schlüssel das Auto öffnet.« Etwas formaler liest sich dieses *Protokoll* so:

MG-Protokoll

AUTOWERK
  für *Autos vom Modell m am Tag d* ***tue***
    $counter \leftarrow 0$
    für *jedes neues Auto a mit Schlüssel s* ***tue***
      $counter \leftarrow counter + 1$
      $a.seriennummer \leftarrow (m, d, counter)$
      $s.seriennummer \leftarrow (m, d, counter)$

AUTO a
  tue ewig
    warte *auf Empfang einer Nachricht m*
    falls $m = (›Öffne‹, a.seriennummer)$ ***dann***
      öffne Türen

AUTOSCHLÜSSEL s
  tue ewig
    falls *›Öffnen‹-Knopf gedrückt* ***dann***
      sende $m = (›Öffne‹, s.seriennummer)$

Aufgabe 0: Auftaktaufgabe

Machen Sie sich den Ablauf des MG-Protokolls an einem Beispiel klar, bei dem Bob und Charly je ein MG-Auto im Parkhaus stehen haben und Bob nun seines mit seinem Schlüssel öffnet. Schreiben Sie die Werte auf, die in den Autos und den Schlüsseln gespeichert sind, und zeichnen Sie ein Diagramm, welche Dinge wo und wie in welcher Reihenfolge in den beiden Autos und in dem Autoschlüssel passieren: »Knopf gedrückt«, »Nachricht m geschickt«, »Nachricht m' empfangen«, »Auto geht auf«, »Auto geht nicht auf« und so weiter.

Sie ahnen sicher schon, dass das MG-Protokoll nicht sonderlich clever ist und vielfältige Angriffe erlaubt. Ein besonders einfacher Angriff nutzt aus, dass die Seriennummern kurz und einfach sind. Sie schicken daher die folgende Analyse an MG, wie Eve das MG-Protokoll angreifen kann, zusammen mit Ihrer Rechnung über 10.000 Euro Beratungshonorar:

Angriff auf das MG-Protokoll

GRUNDIDEE

Schnelles Durchprobieren aller möglichen Seriennummern

VORAUSSETZUNGEN

- Ein Laptop kann pro Millisekunde (ms) eine Funknachricht versenden.
- Ein Auto braucht maximal 1 ms zur Verarbeitung einer Funknachricht.
- Eve muss in der Nähe des Autos sein (mit Laptop).
- Bob sollte nicht in der Nähe des Autos sein.
- Eve kennt Modell und Baujahr des Autos oder kann diese schätzen.

ABLAUF DES ANGRIFFS

input *Modell m des Autos*
input *Baujahr j des Autos*
für $counter \in \{1,2,3,\dots\}$ ***tue***
  für *alle Tage d des Baujahres j* ***tue***
    sende $m = (\text{›}\ddot{O}ffne\text{‹}, (m,d,counter))$

ERFOLGSANALYSE

Eve wird das Auto mit Sicherheit in wenigen Minuten geöffnet haben: Die größten Autowerke der Welt bauen etwa 1000 Autos eines Modells am Tag, so dass Eve pro Sekunde einen Tag des Jahres »durchprobieren« kann.

Aufgabe 1: Angriff auf des Lepo-Protokoll

Die zweite Firma, zu der Sie geschickt werden, heißt Lepo, eine Tochter von MG. Die Chefentwicklerin von Lepo meint, aus den Fehlern von MG gelernt zu haben, und vergibt die Seriennummern nach einem Algorithmus, der viel größere Zahlen erzeugt:

Lepo-Protokoll

AUTOWERK

für *jedes neues Auto a mit Schlüssel s* ***tue***
  $t \leftarrow$ *Anzahl Sekunden seit dem 1. Januar 1900, 00:00:00*
  $c \leftarrow \text{CRYPTOHASH}\big(\text{ALSBITSTRING}(t)\big)$
  $a.seriennummer \leftarrow c$
  $s.seriennummer \leftarrow c$

Das Verhalten der Autos und der Autoschlüssel ist ansonsten wie beim MG-Protokoll. Die Seriennummern sind nun Zahlen mit 256 Bit Länge und Eve kann nun nicht mehr einfach alle 2^{256} Möglichkeiten durchprobieren – egal wie schnell sie Nachrichten verschicken kann. Trotzdem kann Eve ihren Angriff modifizieren für das Lepo-Protokoll und auch Lepo-Autos öffnen, wenn sie lediglich etwas ungestörte Zeit alleine mit dem Auto hat.

Beschreiben Sie den modifizierten Angriff von Eve mit der gleichen Gliederung wie den Angriff auf das MG-Protokoll.

Aufgabe 2: Angriff auf das WV-Protokoll

Die dritte Firma, die Sie beraten, heißt WV. Deren Entwickler nutzen folgendes Protokoll:

WV-Protokoll

AUTOWERK
  für *jedes neues Auto a mit Schlüssel s* ***tue***
    $r \leftarrow$ ERZEUGEZUFALLSZAHL
    $a.seriennummer \leftarrow r$
    $s.seriennummer \leftarrow r$

Wieder ist das Verhalten der Autos und der Schlüssel ansonsten wie beim MG-Protokoll.

Das WV-Protokoll ist deutlich besser und Eve muss nun mehr Aufwand treiben, um das Auto zu öffnen. Natürlich kann sie einfach Bob niederschlagen und seinen Schlüssel klauen oder die Scheiben einschlagen; aber für diese Erkenntnis wird WV Ihnen keine 10.000 Euro Beratungshonorar zahlen. Finden Sie vielmehr einen realistischen Angriff auf das WV-Protokoll durch Eve, bei dem weder Menschen noch Autos zu Schaden kommen. Sie dürfen sich aber aussuchen, was Eve mit ihrem Laptop macht, wann und wo sie dies tut, wie oft sie ihn einsetzt und so weiter.

Beschreiben Sie in der üblichen Gliederung einen realistischen Angriff. Je weniger Voraussetzungen er hat und je größer die Erfolgsaussichten, desto besser.

Aufgabe 3: Angriff auf das WMB-Protokoll

Aufgrund Ihres guten Rufes werden Sie nun von WMB gebeten, deren Protokoll zu analysieren. Die Erzeugung der Seriennummer ist wie im WV-Protokoll; eine Entwicklerin bei WMB ist aber auf die gute Idee gekommen, die Kommunikation zwischen Auto und Bobs Schlüssel weniger vorhersagbar zu machen:

WMB-Protokoll

AUTO *a*
  tue ewig
    warte *auf Empfang der Nachricht ›Öffne‹*
    $r \leftarrow$ ERZEUGEZUFALLSZAHL
    sende *r*
    warte *auf Empfang einer Nachricht m*
    $x \leftarrow m \oplus r$
    falls $x = a.seriennummer$ ***dann***
      öffne Türen

AUTOSCHLÜSSEL *s*
  tue ewig
    falls *›Öffnen‹-Knopf gedrückt* ***dann***
      sende *›Öffne‹*
      warte *auf Empfang einer Zahl r*
      sende $r \oplus s.seriennummer$

Leider hat die Entwicklerin nicht weit genug gedacht. Beschreiben Sie wie zuvor einen realistischen, vielversprechenden Angriff auf das WMB-Protokoll.

Aufgabe 4: Angriff auf das Zneb-Protokoll

Wenig überraschend werden Sie als nächstes von der Firma Zneb, der großen Konkurrenz von WMB, beauftragt. Da die Autos von Zneb echte Luxusautos sind, ist Zneb bereit, die teure CRYPTOHASH Methode in ihre Autoschlüssel einzubauen. Wieder ist die Erzeugung der Seriennummern wie im WV-Protokoll.

Zneb-Protokoll

AUTO a
  $counter \leftarrow 1$
  tue ewig
    warte *auf Empfang einer Nachricht m der Form* $m = (\text{›}\ddot{O}ffne\text{‹}, c, z)$
    $z' \leftarrow \text{CRYPTOHASH}\big(\text{ALSBITSTRING}(c, a.seriennummer)\big)$
    falls $c > counter$ *und* $z = z'$ ***dann***
      $counter \leftarrow c$
      öffne Türen

AUTOSCHLÜSSEL s
  $counter \leftarrow 1$
  tue ewig
    falls *›Öffnen‹-Knopf gedrückt* ***dann***
      $counter \leftarrow counter + 1$
      $z \leftarrow \text{CRYPTOHASH}\big(\text{ALSBITSTRING}(counter, s.seriennummer)\big)$
      sende $(\text{›}\ddot{O}ffne\text{‹}, counter, z)$

Das Protokoll macht es Eve (und Ihnen) nochmal schwerer, einen Angriff durchzuführen – realistische Angriffe sind aber durchaus möglich. Beschreiben Sie einen solchen so wie in den anderen Aufgaben.

Aufgabe 5: Entwicklung des Alset-Protokoll

Nachdem Sie schon bei so vielen Firmen Schwachstellen in den Protokollen aufgedeckt haben, beschließt die Firma Alset, nicht selbst ein Protokoll zu entwickeln, sondern Sie damit zu beauftragen.

Entwickeln Sie selber Protokolle, welche die von Ihnen bei den anderen Herstellern gefundenen Schwachstellen nicht haben. Je weniger Berechnungen vom Autoschlüssel ausgeführt werden müssen, desto besser (das Protokoll von Zneb ist in dieser Beziehung beispielsweise schlechter als die anderen, da hier der Autoschlüssel die teure CRYPTOHASH Methode nutzt). Andererseits ist aber natürlich noch wichtiger, dass die Protokolle sicher sind und keine oder nur extrem unrealistische Angriffe zulassen.

Beschreiben Sie maximal drei Protokolle in derselben Notation wie die anderen Protokolle. Argumentieren Sie klar und nachvollziehbar, weshalb Ihre Protokolle sicher sind.