besprochen, die zum Beispiel bei der Schlüsselgenerierung des RSA-Verfahrens eine Rolle spielen.

Bei der Erzeugung zufälliger Zahlen unterscheiden wir die Erzeugung *echter* Zufallszahlen und die Erzeugung von *Pseudo*-Zufallszahlen. Um echte Zufallszahlen zu erzeugen, braucht man eine physikalische Quelle von Zufall. Zum Beispiel verwendet man hier Detektoren für Teilchen, die in einer gewissen Dichte aus dem Weltall auf die Erde „regnen“. Die gemessene Häufigkeit oder daraus abgeleitete Werte dient dann als Ausgangspunkt für die erzeugte Zufallszahl. Wir werden uns mit solchen Generatoren nicht weiter befassen und verweisen auf [MOV96, Kap. 5.2, 5.6] und [Gut98].

Wenn wir von der Erzeugung von Pseudozufallszahlen (PZZ) sprechen, meinen wir in der Regel die Erzeugung von Zahlen aus einem festen Zahlenbereich, zum Beispiel dem der `unsigned int` mit 32 Bit Länge. Dabei erfolgt die Erzeugung ausgehend von einem vorgegebenen Startwert (engl. **seed**) mit einem deterministischen Algorithmus, der eventuell ebenfalls vom Startwert abhängt. Teilweise werden zur Erzeugung des Seeds echte Zufallszahlen benutzt, man spricht dann von einem *hybriden* Generator.

Die determistisch erzeugten Zahlen sollen von einer zufälligen Folge nicht unterscheidbar sein. Zum Beispiel soll, wenn man eine sehr große Menge solcher erzeugten Zahlen betrachtet, die Häufigkeit jedes Zahlenwertes gleichgroß sein. Zum Test der statistischen Eigenschaften von Pseudo-Zufallszahlen-Generatoren (PZZG) gibt es daher eine ganze Reihe von sogenannten Testsuiten, also Programmen, die statistische Eigenschaften Überprüfen.

Die bekannteste dieser Testsuites ist *Diehard* von G. Marsaglia [Mar]. Daneben hat auch D. Knuth eine Reihe von Tests beschrieben [Knu97] und es gibt die Suite *Test01* von P. L'Ecuyer [LS07]. Schließlich hat das US-amerikanische Standardisierungsinstitut NIST eine Anleitung zum Test von PZZG für *kryptografische* Anwendungen sowie eine Testsuite erstellt [Ruk01].

Ein PZZG in kryptografischen Anwendungen soll nämlich zusätzlich zu den statistischen Eigenschaften seiner Ausgaben noch weitere Eigenschaften aufweisen. So soll er nicht vorhersagbar sein, d. h. man soll aus der Beobachtung einer Reihe von erzeugten Zahlen nicht die nächsten Zahlen vorhersagen oder den inneren Zustand des PZZG ableiten können. Andernfalls wären die nächsten Zufallszahlen, die in einem kryptografischen Protokoll benötigt werden, für einen Angreifer nicht mehr unbekannt und zufällig, sondern bekannt und deterministisch. Weiterhin sollen aus einem (teilweise) kompromittierten Zustand nicht die Vorgängerzustände berechenbar sein.

Ein PZZG ist eigentlich ein endlicher Automat mit Zustand, Ausgabefunktion und Zustandsübergangsfunktion, der zu bestimmten Zeiten durch den Seed in einen bestimmten Zustand versetzt wird und wobei durch den Seed eventuell die Zustandsübergangsfunktion verändert wird. Außerhalb dieser Zeiten erfährt er keine weitere Eingabe. Bei jedem Aufruf, eine PZZ zu erzeugen, ruft der endliche Automat seine Ausgabefunktion auf, die die PZZ erzeugt, und er ruft seine Zustandsübergangsfunktion auf, die den inneren Zustand aktualisiert.

Damit ergeben sich für einen kryptografischen PZZG die Anforderungen, dass die Anzahl der Bits der Ausgabe deutlich geringer sein muss als die Anzahl der Bits des Zustands. Ansonsten könnte aus der Ausgabe auf den Zustand rückgeschlossen werden. Weiterhin muss die Anzahl der Zustände hinreichend groß sein, wobei „hinreichend“ von den Berechnungsmöglichkeiten eines Angreifers abhängt (heute hält man 200 Bit lange Zustände für hinreichend). Denn