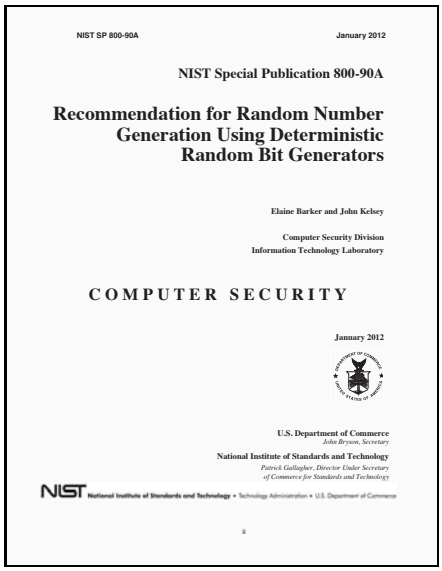
Hintertüren und Schwächen im kryptographischen Standard SP 800-90A

Marc Fischlin


NIST SP 800-90A January 2012

NIST Special Publication 800-90A

Recommendation for Random Number Generation Using Deterministic Random Bit Generators

Elaine Barker and John Kelsey

Computer Security Division
Information Technology Laboratory

COMPUTER SECURITY

January 2012

U.S. Department of Commerce
John Bryson, Secretary

National Institute of Standards and Technology
Patrick Gallagher, Director Under Secretary of Commerce for Standards and Technology

NIST National Institute of Standards and Technology • Technology Administration • U.S. Department of Commerce

Die durch Edward Snowden angestoßenen Enthüllungen liefern seit Juni 2013 immer mehr bemerkenswerte Details darüber, welche Methoden Geheimdienste zur Sammlung und Verwertung von personenbezogenen Daten anwenden. Im vorliegenden Artikel diskutieren wir speziell den Fall des kryptographischen Standards SP 800-90, bei dem der amerikanische Geheimdienst NSA (National Security Agency) gemäß eines Artikels der New York Times vorsätzlich Schwächen eingebaut hatte. Wir erläutern anhand der mathematischen Details, wie diese Hintertür aussieht und wie die Schwächen einzuordnen sind.

1 Kryptographische Standards

Technische Standards dienen der Vereinheitlichung von Normen und Verfahren und sollen dadurch die Interoperabilität fördern. Auch in der Kryptographie gibt es zahlreiche Standards und Standardisierungsorganisationen. Eine wichtige Institution ist hier neben der International Organization for Standardization (ISO) und der International Electrotechnical Commission (IEC) das amerikanische National Institute of Standards and Technology (NIST). NIST setzt mit seiner Reihe „Special Publications (800 series)“ Standards speziell für Themen zur Informationssicherheit (siehe [10]). So beschreiben beispielsweise die Standards SP 800-56A bis SP 800-56C, wie zwei Teilnehmer basierend auf dem Diskreten Logarithmus- oder dem Faktorisierungsproblem einen gemeinsamen kryptographischen Schlüssel aushandeln können.

Die von NIST zunächst nur für die Vereinigten Staaten entwickelten Standards etablieren sich auch oft zu internationalen de-facto-Standards und werden teilweise in ISO/IEC-Standards übernommen. Dabei ist bemerkenswert, dass die Standards im Bereich der IT-Sicherheit oft ein weiteres Attribut zugewiesen bekommen, nämlich, dass standardisierte Verfahren auch besondere Sicherheitsgarantien mit sich bringen. Die Erfahrung hat allerdings gezeigt, dass diese Annahme im Allgemeinen zu optimisitisch ist. Dies gilt um so mehr, als wir heute wissen, dass Geheimdienste absichtlich schwache Standards vorgeschlagen haben. Wir betrachten hier mit dem Standard SP 800-90 der NIST einen solchen Fall.

2 Schwächen im Standard SP 800-90

Der Standard SP 800-90 beschäftigt sich mit der Erzeugung von sogenannten Pseudozufallszahlen, also quasi zufälligen Werten. Dieses Konzept wird ausführlicher in Abschnitt 2.2 diskutiert. Der NIST-Standard SP 800-90 bzw. der ergänzende Standard SP 800-90A diskutiert Möglichkeiten, wie man solche Pseudozufallswerte auf vermeintliche sichere Weise generieren kann. Teile des NIST-Standards finden sich auch im ISO/IEC Standard 18031:2011 [8] wieder.

2.1 Was war passiert?

Schon kurz nach Veröffentlichung der ersten Version des Standards SP 800-90 im Jahr 2006 – die Version SP 800-90A ergänzte den ursprünglichen Standard und die aktuelle Version datiert auf Januar 2012 – wurde Kritik am Standard öffentlich. Wissenschaftliche Arbeiten [2, 15, 17] aus den Jahren 2006 und 2007 zeigten, dass eines der vorgeschlagenen Verfahren, der Pseudozufallsgenerator Dual_EC_DRBG, nicht die gewohnten Sicherheitsstandards erfüllte. Schon damals ließen Kommentatoren wie Bruce Schneier anklingen, dass es sich dabei um absichtlich implementierte Schwächen handeln könnte [14]. Tatsächlich bestätigt der aktuelle Standard SP 800-90A die Mitarbeit der NSA an der Erstellung, wobei die Zusammenarbeit mit den IT-Sicherheitsexperten der NSA an solchen Standards *per se* nicht ungewöhnlich ist. Erst im Jahr 2013 wurde im Rahmen der Enthüllungen um Edward Snowden durch einen Artikel der New York

 DOI 10.1515/dmvm-2014-0012

Times [12] bestätigt, dass die NSA Schwächen absichtlich eingebaut hat.

Inzwischen rät NIST selbst explizit von der Verwendung des Generators ab [11]. In einigen kommerziellen Produkten wurde der Generator implementiert, aber bis auf zwei Ausnahmen, dem BSAFE-Toolkit und dem Data Protection Manager der Firma RSA, nicht als der standardmäßig verwendete Generator eingesetzt, sondern nur als Option [3]. Der Generator sollte natürlich nicht mehr verwendet werden. RSA selbst empfiehlt inzwischen, in dem betroffenen Produkt auf einen anderen Generator umzusteigen [4]. Bemerkenswert ist, dass beide Empfehlungen, sowohl die von NIST als auch die von RSA, erst nach der Veröffentlichung des Artikels der New York Times im September 2013 herausgegeben wurden, obwohl die ersten Kritiken an der kryptographischen Sicherheit des Dual_EC_DRBG bereits 2006 und 2007 veröffentlicht wurden.

2.2 *Hintergrund: Pseudozufallsgeneratoren*

Zufallswerte sind essenziell für die Sicherheit kryptographischer Verfahren. Sie treten an zahlreichen Stellen in Verfahren auf, angefangen mit der Erzeugung der geheimen Schlüssel, über die Wahl von Initialisierungsvektoren für Verschlüsselungen, bis hin zur Generierung von unvorhersagbaren Fragen in Challenge-Response-Verfahren zur Authentisierung. Klassische Rechner können aber in der Regel keine „echten“ Zufallswerte erzeugen, da sie deterministisch arbeiten, also ihre Ergebnisse eindeutig vorherbestimmt sind. Die folgende Aussage, die dem Mathematiker und Informatiker John von Neumann zugeordnet wird, fasst dieses Dilemma pointiert zusammen:

> Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin. (John von Neumann)

Der Entropie-Begriff von Shannon [16], den er in seiner wegweisenden Arbeit von 1948 über Informationstheorie einführte, bestätigt diese Sichtweise. Für eine (diskrete) Zufallsvariable X ist die Shannon-Entropie definiert als

$$H(X) := -\sum_x \mathsf{Prob}[X = x] \cdot \log_2 \mathsf{Prob}[X = x],$$

wobei x die möglichen Werte von X durchläuft. Intuitiv gibt die Shannon-Entropie an, wieviele Bits man durchschnittlich benötigt, um den Ausgang einer Stichprobe von X zu kommunizieren. Die Shannon-Entropie ist genau dann maximal, wenn X uniform verteilt ist.

Für jede Zufallsvariable und jede mathematische Funktion C (die der Leser hier als einen deterministischen Computer ansehen kann) gilt

$$H(X) \geq H(C(X)),$$

wobei $C(X)$ die Zufallsvariable beschreibt, die zuerst eine Stichprobe von X wählt und dann C anwendet. Folglich kann kein (deterministischer) Computer im Sinne der Shannon-Entropie „zusätzlichen Zufall“ erzeugen, auch wenn man ihm „etwas Zufall“ in Form von X zur Verfügung stellt.

Die Kryptographie löst das Dilemma, indem sie die Anforderung an die Zufallswerte abschwächt, und lediglich *pseudozufällige* Werte verlangt. Pseudozufällige Werte erreichen zwar keine maximale Shannon-Entropie, sehen aber dennoch für alle praktischen Zwecke wie echt zufällig aus. In der Kryptographie wird dies formal in die Theorie sogenannter *Pseudozufallsgeneratoren* eingebettet. Ein Pseudozufallsgenerator G ist ein deterministischer (und effizienter) Algorithmus, der eine kurze, echt zufällige Eingabe – üblicherweise als binäre Zeichenkette r aus der Menge $\{0,1\}^*$ aller endlichen Bitfolgen kodiert – in eine längere Ausgabe transformiert, sodass diese Ausgabe „wie zufällig“ aussieht. Letzteres wird mit Hilfe des Begriffs der Ununterscheidbarkeit formalisiert, soll hier aber nicht weiter ausgeführt werden. Eine Einführung gibt das Buch von Goldreich [6].

In der Notation der Entropie verlangt man, dass ein Pseudozufallsgenerator maximale *Pseudo-Entropie* [7] besitzt. Eine Zufallsvariable X hat Pseudo-Entropie

$$H^{\mathsf{HILL}}(X) \geq k,$$

wenn sie ununterscheidbar (im obigen Sinne) von einer Zufallsvariablen Y mit Shannon-Entropie $H(Y) \geq k$ ist. Folglich hat ein Pseudozufallsgenerator, der für n-Bit-Eingaben längere $\ell(n)$-Bit-Ausgaben erzeugt, Pseudo-Entropie $\ell(n)$, während die Shannon-Entropie maximal n sein kann. Für die meisten kryptographischen Anwendungen genügt nachweislich eine hohe Pseudo-Entropie, um Sicherheit zu gewährleisten.

Ob sichere Pseudozufallsgeneratoren existieren, lässt sich mit gegenwärtigen Methoden nicht nachweisen. Man kann solche Generatoren aber unter sehr schwachen kryptographischen Annahmen ableiten [7], unter anderem auch unter gängigen Sicherheitsannahmen wie der Schwierigkeit, die diskrete Exponentiation effizient zu invertieren (dem sogenannten Diskreten-Logarithmus-Problem). Daher gilt es im Augenblick als vernünftig, die Existenz solcher Generatoren anzunehmen. Gleichzeitig muss man aber besondere Vorsicht walten lassen, wenn man einen Generator entwickelt.

Wie setzen moderne Computer nun solche Pseudozufallsgeneratoren ein? Üblicherweise wird der Computer mit einem kurzen, möglichst zufälligen Wert initialisiert, beispielsweise durch Messungen von rauschanfälligen Komponenten während des Starts, oder durch „zufällige“ Eingaben des Anwenders. Bei Bedarf erzeugt der Generator aus seinem Startwert hinreichend viele Pseudozufallsbits, gibt einen Teil als Ausgabe aus, und aktualisiert gleichzeitig seinen Startwert mit einem anderen Teil der generierten Bits. Die tatsächlichen Verfahren variieren dieses Schema auf vielfältige Weise.

2.3 Der Generator Dual_EC_DRBG

Der Standard SP 800-90 beschreibt verschiedene, vermeintlich sichere Pseudozufallsgeneratoren. Eine Variante ist der Generator Dual_EC_DRBG, der auf elliptischen Kurven beruht.

Elliptische Kurven werden wegen ihrer guten Sicherheitseigenschaften – kurze Schlüssel und Resistenz gegen bekannte Diskrete Logarithmus-Verfahren – verstärkt in der Kryptographie eingesetzt. Im vorliegenden Fall wurden im Standard eine Primzahl p sowie die Konstanten a und b spezifiziert (z. B. ist $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} + 1$), die eine elliptische Kurve definieren. Die Kurvenpunkte

$$E_{a,b}(\mathbb{F}_p) = \left\{(x, y) \in \mathbb{F}_p^2 \mid y^2 = x^3 + ax + b\right\},$$

bilden zusammen mit einer entsprechend definierten Addition eine Gruppe, die für die Werte hier sogar von primer Ordnung ist. Die x-Koordinaten von Kurvenpunkten können nach Wahl von p mit 256 Bits dargestellt werden. Der Standard spezifiziert zusätzlich zwei Punkte $P = (x_P, y_P)$ und $Q = (x_Q, y_Q)$, die beide verschieden vom „Punkt im Unendlichen“ $\mathcal{O}$, dem neutralen Element der Gruppe, sind. Der Punkt P ist ein Erzeuger der additiven Gruppe.

Der Dual_EC_DRBG wird mit einem zufälligen Startwert s_0 aus $\mathbb{F}_p$ initialisiert. Bei Bedarf generiert das Verfahren 240 Pseudozufallsbits, indem es ausgehend vom aktuellen Wert $s_i \in \mathbb{F}_p$ den Kurvenpunkt $s_i P$ berechnet und s_{i+1} als die x-Koordinate $\text{x-coord}(s_i P)$ des Punktes für die nächste Iteration speichert. Ebenso berechnet das Verfahren den Kurvenpunkt $s_{i+1} Q$ und gibt nun die untersten 240 Bits der 256 Bits der x-Koordinate des Punktes (in einer üblichen Binärdarstellung) als Pseudozufallsbits aus. Benötigt man mehr Bits, wiederholt man das Verfahren hinreichend oft.

Das Design des Generators beruht auf dem sogenannten *Diffie-Hellman-Entscheidungsproblem*. Beim bekannten Schlüsselaustauschverfahren von Diffie und Hellman [5] erzeugen die zwei Teilnehmer Alice und Bob einen gemeinsamen geheimen Schlüssel in einer elliptischen Kurve mit Erzeuger P, indem Alice für zufälliges q einen Punkt $Q = qP$ berechnet und Q an Bob sendet, und Bob $R = rP$ für zufälliges r berechnet, und mit R antwortet. Alice kann nun lokal $S = qR = (qr)P$ berechnen, und Bob ebenfalls $S = rQ = r(qP) = (qr)P$, sodass S der gemeinsame Schlüssel wird.

Das klassische *Diffie-Hellman-Berechnungsproblem*, auf dessen Sicherheit das Diffie-Hellman-Verfahren beruht, besagt nun, dass es für einen Angreifer schwierig ist, ebenfalls den Schlüssel S aus den ausgetauschten Daten P, Q, R zu berechnen. Das DH-Entscheidungsproblem fordert nun sogar, dass man den Schlüssel S nicht einmal von einem unabhängig gewählten Kurvenpunkt unterscheiden kann: Gegeben vier Kurvenpunkte (P, Q, R, S) entscheide man, ob S ein unabhängiger und uniform verteilter Punkt ist, oder ob $S = rQ$ für $R = rP$ gilt. Beide Probleme, das Berechnungsproblem und das Entscheidungsproblem, gelten in der Kryptographie als schwierig, und das DH-Verfahren somit als sicher. Weitere Informationen zu den beiden Problemen findet man in [1].

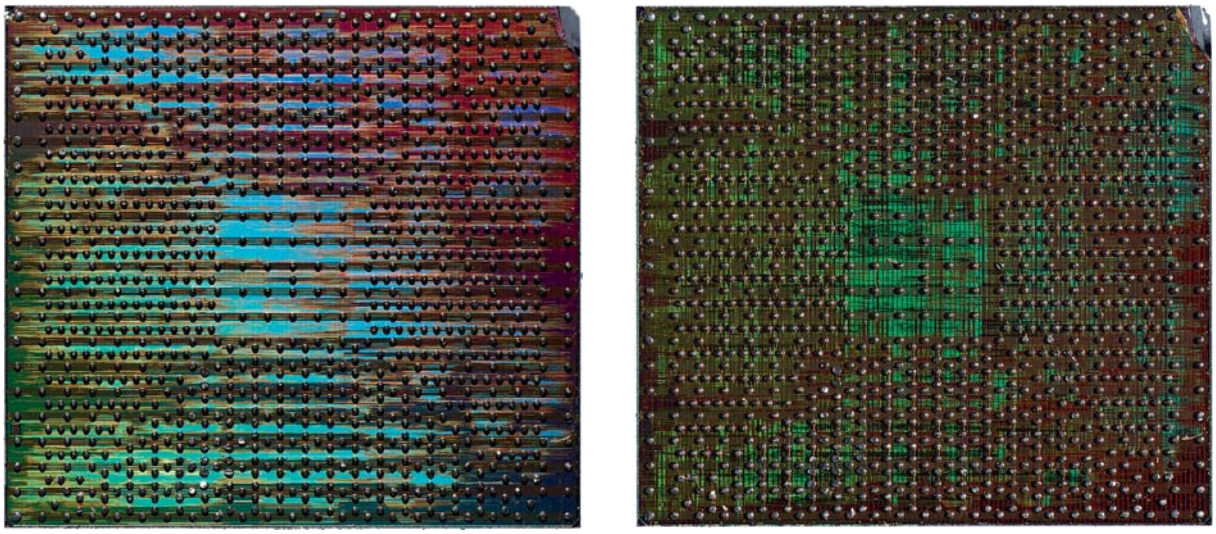
Im Fall des Generators Dual_EC_DRBG spielt $s_{i+1} Q$ die Rolle der Schlüssels S, der wie zufällig aussieht, selbst wenn man P, Q und sogar den in der nächsten Iteration des Generators berechneten Wert $R = s_{i+1} P$ kennen würde. In diesem Sinne ist das *grundsätzliche* Design-Prinzip des Generators plausibel, auch wenn jede Iteration durch zwei Multiplikationen in der elliptischen Kurve wesentlich langsamer als andere bekannte Generatoren ist. Dass der Generator dennoch Schwächen hat, liegt an den Details, wie wir im folgenden Abschnitt diskutieren.

2.4 Schwächen im Generator Dual_EC_DRBG

Bereits kurz nach Erscheinen des Standards haben die ersten Analysen [2, 15] gezeigt, dass die Ausgaben des Dual_EC_DRBG nicht die übliche kryptographische Sicherheit bieten. Diese Analysen nutzen aus, dass die Punkte der elliptischen Kurve bei Projektion auf die untersten 240 Bits der x-Koordinate eben nicht uniform sind: Ein nicht zu vernachlässigender Anteil der Menge der Zeichenketten aus 240 Bits kann nicht von solchen Punkten getroffen werden. Dieser Schritt, die einfache Projektion auf die untersten Bits, schleust also Schwächen ein, obwohl die Kurvenpunkte uniform verteilt sind. Es ist bemerkenswert, dass hier schon vor Veröffentlichung des Standards bessere Verfahren zur Glättung der Verteilung bekannt waren (beispielsweise in [7]).

Die Autoren von [2, 15] zeigen, dass sie wegen der einfachen Projektion auf die Bits die Ausgabe des Dual_EC_DRBG mit einem Vorteil von ca. 0,1 % gegenüber echt zufälligen Werten erkennen können. Indem sie mehrere Ausgaben untersuchen, lässt sich dieser Vorteil sogar auf einige Prozentpunkte steigern. Anders ausgedrückt: Die Pseudo-Entropie des Dual_EC_DRBG ist nicht maximal. Zwar gelten Generatoren mit solchen verzerrten Ausgaben in der Kryptographie nicht als sicher, dennoch war nicht bekannt, wie man diese Schwäche des Dual_EC_DRBG unmittelbar hätte ausnutzen können, um beispielsweise große Teile eines geheimen Schlüssels verlässlich zu berechnen. Wie dies gehen könnte, zeigte die Arbeit von Shumov und Ferguson [17]. Die beiden Autoren bestätigen, dass man bei geschickter Wahl der Punkte P und Q – was für die Entwickler des Standards durchaus möglich gewesen wäre – die Ausgaben des Generators mit geringem Aufwand vorhersagen kann.

Der Angriff von Shumov und Ferguson basiert auf folgender Beobachtung. Da P ein Erzeuger, $Q \neq \mathcal{O}$ und die Ordnung der elliptischen Kurve prim ist, gibt es ein ganzzahliges e mit $P = eQ$. Shumov und Ferguson gehen nun davon aus, dass der Angreifer den Wert e kennt. Der Angreifer betrachtet dann eine Ausgabe $r \in \{0,1\}^{240}$ einer Iteration des Generators, die er beispielsweise als Teil eines Initialisierungsvektors einer Verschlüsselung oder als



Die deterministische Chip-Sicht: Ein G5 in unterschiedlich einfallendem Licht (Sammlung Günter M. Ziegler. Foto: Christoph Eyrich)

Frage in einer Challenge-Response-Authentisierung lernen kann. Er berechnet alle möglichen $2^{16} = 65.536$ Bitfolgen der Länge 256, die auf r enden. Für jeden dieser Werte testet der Angreifer, ob diese Bitfolge der x-Koordinate eines Punktes auf der elliptischen Kurve entspricht, indem er prüft, ob $z = x^3 + ax + b \bmod p$ ein quadratischer Rest in $\mathbb{F}_p$ ist. Jedes solche x gibt maximal zwei mögliche Werte für y. Der Gesamtaufwand dafür ist so gering, dass er mit jedem herkömmlichen Rechner zu bewerkstelligen ist.

Der Angreifer erhält so eine Menge von maximal $2 \cdot 2^{16}$ Kurvenpunkten $R_1, R_2, R_3, \ldots$, wobei der „richtige" Punkt $s_{i+1}Q$ darin enthalten sein muss. Mit Hilfe der Kenntnis von e kann der Angreifer dann die Punkte $eR_1, eR_2, eR_3, \ldots$ berechnen, sodass darunter auch der Punkt $s_{i+1}P = s_{i+1}(eQ) = e(s_{i+1}Q)$ sein muss. Folglich kennt der Angreifer eine beschränkte Menge von Punkten, deren x-Koordinate auch den übernächsten Zustandswert $s_{i+2} = \text{x-coord}(s_{i+1}P)$ des Generators enthält. Würde er den Punkt eindeutig kennen, könnte der Angreifer alle folgenden Ausgaben des Generators dann exakt vorhersagen! Dazu muss er aber nun lediglich einige weitere Ausgaben des Generators beobachten, um die Menge der potenziellen Kandidaten auf ein Element zu reduzieren. Folglich ist der Generator bei Kenntnis von e mit $P = eQ$ vollkommen unsicher.

Der Angriff von Shumov und Ferguson ist nicht überraschend, wenn man bedenkt, dass das DH-Berechnungsproblem und auch das DH-Entscheidungsproblem bei bekanntem e mit $P = Qe$ beide leicht zu lösen sind: Gegeben $P, Q = qP, R$ und eben e, das multiplikative Inverse zu q modulo der bekannten Gruppenordnung, kann man leicht den Schlüssel $S = rQ = (rq)P$ für $R = rP$ berechnen, indem man einfach q aus e berechnet und $S = qR$ bildet. In diesem Sinne „spielt" der Angreifer dann quasi die Rolle von Alice im DH-Schlüsselaustausch. Nur wenn Q nachweislich so bestimmt worden wäre, dass niemand effizient e berechen kann, wäre die Sicherheit des Dual_EC_DRBG gewährleistet. Dass dies nicht der Fall war, hat die Veröffentlichung der New York Times gezeigt.

2.5 Wie geht es weiter?

Mehr als fünf Jahre nach Bekanntwerden der Schwächen, aber erst nach Bestätigung durch die New York Times, dass die NSA die Spezifikation beeinflusst hat, raten inzwischen sowohl NIST als auch Firmen wie RSA von der Verwendung des Generators Dual_EC_DRBG ab. Man kann daher hoffen, dass in Zukunft keine Angriffe über diese Schwachstelle erfolgen werden.

In einem im Dezember 2013 erschienenen Artikel der Nachrichtenagentur Reuters [9] wurde behauptet, dass die Firma RSA für die Verwendung des schwachen Generators Dual_EC_DRBG in ihrem BSAFE-Produkt von der NSA bezahlt wurde. Die Firma bestritt diese Vorwürfe umgehend [13]. Eine Klärung steht noch aus.

NIST hat inzwischen den Standard SP 800-90A auf den Status eines „Draft" zurückgesetzt und im Zeitraum von September 2013 bis November 2013 um Kommentierung gebeten. Das weitere Vorgehen der NIST ist zum Zeitpunkt, zu dem dieser Artikel verfasst wurde, nicht bekannt.

3 Fazit

Die Rolle der NSA bei der Erstellung des Standards SP 800-90, als auch die weiteren bekannt gewordenen Maßnahmen der Geheimdienste, sind natürlich äußerst bedenklich. Im Fall des Dual_EC_DRBG wurden die eingebauten Schwächen unmittelbar entdeckt. Dennoch

blieb der Generator Teil des Standards und wurde auch in Produkten verwendet. Diese Tatsache zeigt – unabhängig davon, ob Standards bewusst oder unbewusst eingebaute Schwächen beinhalten – das Problem heutiger Sicherheitsstandards. Sie beruhen in der Regel zwar auf bekannten Sicherheitsprinzipien, fallen aber in ihren Sicherheitsbetrachtungen gegenüber Analysen, wie sie in akademischen Veröffentlichungen heute üblich sind, stark ab. Solche wissenschaftlichen Betrachtungen schließen in der Regel sogenannte Reduktionsbeweise oder ausführliche kryptanalytische Techniken ein, während die Sicherheitskriterien von Standards im Allgemeinen vage bleiben. Trotzdem genießen standardisierte Verfahren einen verblüffenden Vertrauensvorschuss bezüglich ihrer Sicherheit, und werden bedenkenlos in Produkten verwendet. Hier sind allerdings nicht nur die Entwickler von Standards und die Firmen gefordert, sondern auch Kryptographen, um gemeinsam die Sicherheitsgarantien von kryptographischen Standards zu verbessern.

Literatur

[1] Dan Boneh. The decision Diffie-Hellman problem. volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer, 1998.

[2] Daniel R. L. Brown and Kristian Gjøsteen. A security analysis of the NIST SP 800-90 elliptic curve random number generator. In *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 466–481. Springer, 2007.

[3] Computer Emergency Response Team (CERT). Dual_EC_DRBG output using untrusted curve constants may be predictable. Vulnerability Note VU#274923, November 2013.

[4] Computer Emergency Response Team (CERT). Dual_EC_DRBG output using untrusted curve constants may be predictable. RSA Security, Inc. Information to Vulnerability Note VU#274923, September 2013.

[5] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[6] Oded Goldreich. *Pseudorandom Generators: A Primer*. ULECT series. AMS, 2010.

[7] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[8] Internationan Organization for Standardization. Iso/iec 18031:2011: Information technology – security techniques – random bit generation, 2011.

[9] Joseph Menn. Exclusive: Secret contract tied NSA and security industry pioneer. Reuters, Dezember 2013.

[10] National Institute of Standards and Technology. Special publications (800 series). http://csrc.nist.gov/publications/PubsSPs.html, 2013.

[11] National Institute of Standards and Technology. Supplemental ITL bulletin for September 2013, September 2013.

[12] Nicole Perlroth. Government announces steps to restore confidence on encryption standards. New York Times, September 2013.

[13] RSA, The Security Division of EMC. RSA response to media claims regarding NSA relationship. https://blogs.rsa.com/news-media-2/rsa-response/, Dezember 2013.

[14] Bruce Schneier. The strange story of Dual_EC_DRBG. www.schneier.com/blog, November 2007.

[15] Berry Schoenmakers and Andrey Sidorenko. Cryptanalysis of the dual elliptic curve pseudorandom generator. *IACR Cryptology ePrint Archive*, 190, 2006.

[16] Claude Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948.

[17] Dan Shumow and Niels Ferguson. On the possibility of a back door in the NIST SP 800-90 Dual EC PRNG. Crypto Rump Session, 2007.

Prof. Dr. Marc Fischlin, Fachbereich Informatik,
TU Darmstadt, Mornewegstraße 30, 64293 Darmstadt
marc.fischlin@cryptoplexity.de, www.cryptoplexity.de

Marc Fischlin ist Heisenberg-Professor für Kryptographie und Komplexitätstheorie an der Technischen Universität Darmstadt. Er beschäftigt sich primär mit Sicherheitsbeweisen zu kryprographischen Protokollen. Ein Aspekt seiner Tätigkeit umfasst die Beurteilung standardisierter Verfahren, auch in Zusammenarbeit mit öffentlichen oder privatwirtschaftlichen Einrichtungen.