Journal of Cryptology **12** (1999), 185–192.

# HOW TO STRETCH RANDOM FUNCTIONS: THE SECURITY OF PROTECTED COUNTER SUMS

DANIEL J. BERNSTEIN

ABSTRACT. Let $f$ be an unpredictable random function taking $(b+c)$-bit inputs to $b$-bit outputs. This paper presents an unpredictable random function $f'$ taking variable-length inputs to $b$-bit outputs. This construction has several advantages over chaining, which was proven unpredictable by Bellare, Kilian, and Rogaway, and cascading, which was proven unpredictable by Bellare, Canetti, and Krawczyk. The highlight here is a very simple proof of security.

## 1. INTRODUCTION

When $k$ is kept secret, the function $\mathrm{surf}_k$ defined in [9], taking 384-bit inputs to 256-bit outputs, appears to be unpredictable. There is a \$1000 reward for anyone who can predict $\mathrm{surf}_k$.

Starting from $\mathrm{surf}_k$ one can construct efficient secret-key solutions to a variety of cryptographic problems; see, for example, [8] and [5]. These solutions are provably secure, in the sense that anyone who can break them can also break $\mathrm{surf}_k$, and collect the \$1000 reward, in a few extra steps. Different solutions compete on the exact meaning they give to "efficient" and "few."

In this paper, I introduce the "protected counter sum" construction. Given a function $f$ from 384-bit inputs to 256-bit outputs, I construct a function $f'$ from variable-length inputs to 256-bit outputs. I show that if $f$ is an unpredictable random function then $f'$ is also unpredictable. See section 4. This construction compares favorably with chaining, which was proven unpredictable in [7], and cascading, which was proven unpredictable in [3].

All the ideas in the protected counter sum construction are already present in [5] and [3]. My main contribution is the exact security analysis, specifically Theorem 3.1.

**Implementation.** I wrote a portable C library for protected counter sums, using $\mathrm{surf}_k$ as the underlying random function. The library, compiled with `gcc` 2.6 on a Pentium, occupies 716 bytes. It uses approximately 600 Pentium cycles per $\mathrm{surf}_k$ call for byte-to-word conversion and other housekeeping. The total time for a short message (up to one 256-bit block) is 6219 Pentium cycles. The total time for a long message is 3043 cycles per input block; this is 8.4 million bits per second on a Pentium-100.

---

*Date*: 19971230.

1991 *Mathematics Subject Classification.* Primary 94A60.

*Key words and phrases.* unpredictable random functions, variable-length inputs, exact security analysis.

The author was supported by the National Science Foundation under grant DMS–9600083.

**Terminology.** This paper uses standard terminology from probability theory. See the appendix for the definition of "random." Warning for readers unfamiliar with probability theory: random variables are not necessarily uniformly distributed.

An **oracle algorithm** is an algorithm that uses an oracle.

## 2. Unpredictable random functions

Fix sets $S$ and $T$, with $T$ finite. Let $A$ be an oracle algorithm that prints either 0 or 1. A function $h : S \to T$ can be viewed as an oracle that prints $h(x)$ given $x$. Define $A(h) \in \{0, 1\}$ as the result of running $A$ with this oracle.

Let $f$ and $g$ be random functions from $S$ to $T$, i.e., random variables all of whose values are functions from $S$ to $T$. The **$A$-distance between $f$ and $g$** is $|\Pr[A(f) = 1] - \Pr[A(g) = 1]|$. If the $A$-distance between $f$ and $g$ is negligible for every practical algorithm $A$ then $f$ and $g$ have **indistinguishable distributions**. (This concept is parametrized by the notions of "negligible" and "practical.")

In particular, let $g$ be a uniform random function from $S$ to $T$. A random function $f : S \to T$ is **unpredictable** if $f$ and $g$ have indistinguishable distributions.

**Examples.** A uniform random function is unpredictable.

A uniform random constant function from $S$ to $T$ is predictable, if $\#S > 1$ and $\#T > 1$. Let $k$ be a uniform random element of $T$, and consider the function $x \mapsto k$ from $S$ to $T$. Let $A$ be an oracle algorithm that feeds two distinct inputs to the oracle and prints 1 iff the outputs are the same. Then the $A$-distance is $1 - 1/\#T$.

When $k$ is a uniform random 1024-bit string, the random function $\mathrm{surf}_k$ defined in [9] seems to be unpredictable: there are no known practical algorithms that predict $\mathrm{surf}_k$ with any noticeable probability. However, my ignorance does not constitute a proof. Perhaps every easily computed low-entropy random function is predictable.

**Generalizations.** For any measurable space and any notion of an oracle on that space, one can consider distinguishability, using that oracle, of random elements of that space.

For example, a uniform random constant function is unpredictable with a *one-time oracle*, i.e., an oracle that will answer only one question. One-time security is not sufficient for most practical applications.

The following random function on a finite field is unpredictable with an $n$-time oracle; see [17, page 486] and [11]. Select independent uniform random elements $k_0, k_1, \ldots, k_{n-1}$ in the field, and consider the function $x \mapsto k_0 + k_1 x + \cdots + k_{n-1} x^{n-1}$.

**Notes.** In [29], while introducing the Turing test, Turing discussed the claim that an observer of mechanical behavior could always figure out the underlying rules of behavior. He pointed out that a particular easily computable low-entropy random function seemed to be unpredictable. Unfortunately the details of Turing's example were never published.

A specific asymptotic form of Turing's notion of unpredictability was introduced by Goldreich, Goldwasser, and Micali in [15], and studied further by Luby and Rackoff in [19]. The theorems stated in [15] and [19], being purely asymptotic, are irrelevant to applied cryptography, though the constructions underlying the theorems are useful.

Several recent papers by Bellare et al., including [7], [5], and [3], continue Turing's concrete study of unpredictability, taking constant factors and practical issues into account.

"Unpredictable" has several aliases in the literature: "cryptographically strong" (see [26] or [15]), "cryptographically secure" (see [20]), and "pseudorandom" (see, e.g., [19]). "Fixed-input/variable-input pseudorandom function" is used in [3] where I would say "unpredictable random function on blocks/messages" respectively. I find it distasteful to use "pseudorandom" to mean "passes all statistical tests"; for fifty years the standard meaning of "pseudorandom" has been "passes *some* statistical tests." See [16], [10], [22], [20], or [21, page 950].

## 3. Proving information-theoretic unpredictability

Let $S$ and $T$ be finite sets. Let $q_1, q_2, \ldots, q_m$ be distinct elements of $S$. A random function $f$ from $S$ to $T$ is **within $\epsilon$ of uniform on** $\{q_1, q_2, \ldots, q_m\}$ if

$$\Pr[f(q_1) = r_1, f(q_2) = r_2, \ldots, f(q_m) = r_m] \geq \frac{1-\epsilon}{\#T^m}$$

for every $(r_1, r_2, \ldots, r_m) \in T^m$.

**Theorem 3.1.** *Let $f$ be a random function from $S$ to $T$. Assume that $f$ is within $\epsilon$ of uniform on every set of size at most $n$. Let $A$ be an oracle algorithm that performs at most $n$ distinct oracle queries. Then the $A$-distance between $f$ and uniform is at most $\epsilon$.*

The run time of $A$ is irrelevant.

*Proof.* Write $P_f(E)$ for the probability of $E$ if $A$ uses $f$ as an oracle. Let $g$ be a uniform random function from $S$ to $T$. This proof compares $P_f(E)$ to $P_g(E)$ for various events $E$.

Fix $q = (q_1, q_2, \ldots, q_m) \in S^m$ and $r = (r_1, r_2, \ldots, r_m) \in T^m$ with $q_1, q_2, \ldots, q_m$ distinct and $m \leq n$. Define $X_{q,r}$ as the event that $h(q_j) = r_j$ for each $j$, where $h$ is the oracle used by $A$. By hypothesis $P_f(X_{q,r}) \geq (1-\epsilon)/\#T^m = (1-\epsilon)P_g(X_{q,r})$.

Next define $Y_{q,r}$ as the event that (1) $A$ prints 1; (2) $A$ performs exactly $m$ distinct oracle queries; (3) for each $j$, the $j$th distinct query from $A$ is $q_j$—i.e., $q_1$ is the first query, $q_2$ is the first query different from $q_1$, etc.; and (4) for each $j$, the oracle's reply to $q_j$ is $r_j$.

The conditional probability of $Y_{q,r}$ given $X_{q,r}$ is predetermined by $A$; it does not depend on $h$. Indeed, it is the chance that $A$ decides to do $q_1$, to do $q_2$ given $(q_1, r_1)$, to do $q_3$ given $(q_1, r_1, q_2, r_2)$, etc., and finally to stop and print 1 given $(q_1, r_1, \ldots, q_m, r_m)$. Thus $P_f(Y_{q,r}) \geq (1-\epsilon)P_g(Y_{q,r})$.

The probability that $A$ prints 1 is the sum of the $Y_{q,r}$ probabilities over all possible $(q, r)$. Conclusion: $\Pr[A(f) = 1] \geq (1-\epsilon)\Pr[A(g) = 1] \geq \Pr[A(g) = 1] - \epsilon$. Similarly $\Pr[A(f) \neq 1] \geq \Pr[A(g) \neq 1] - \epsilon$. Thus the $A$-distance between $f$ and $g$ is at most $\epsilon$. □

Theorem 3.1 can be generalized in several ways. It is not necessary to assume that $S$ is finite. It is also not necessary to limit the number of queries from $A$, as long as $f$ is within $\epsilon$ of uniform on every possible set of queries.

## 4. Protected counter sums

Let $f$ be a function from $(b+c)$-bit blocks to $b$-bit blocks. Let $(p_1, p_2, \ldots, p_k)$ be a sequence of $b$-bit blocks, of length $k$ between 0 and $2^c - 1$ inclusive. I define $f^+(p_1, p_2, \ldots, p_k)$ as the **counter sum** $f(\underline{1}, p_1) + f(\underline{2}, p_2) + \cdots + f(\underline{k}, p_k)$, and $f'(p_1, p_2, \ldots, p_k)$ as the **protected counter sum** $f(\underline{0}, f^+(p_1, p_2, \ldots, p_k))$. Here $\underline{i}$ means any convenient encoding of $i$ into $c$ bits, and $+$ is a convenient group operation on $b$-bit blocks, such as exclusive-or.

The counters $\underline{1}, \ldots, \underline{k}$ hide input patterns. The sum $f^+(p_1, \ldots, p_k)$ is predictable from its linear structure—if $p_1$ is changed, the output difference is independent of $p_2$—but it is protected inside $f(\underline{0}, \cdot)$, so an attacker cannot recognize output differences other than 0.

Let $A$ be an oracle algorithm. Consider the following oracle algorithm $A'$: run $A$, answering a query for $(p_1, p_2, \ldots, p_k)$ with $h(\underline{0}, h(\underline{1}, p_1) + h(\underline{2}, p_2) + \cdots + h(\underline{k}, p_k))$, where $h$ is the oracle for $A'$. Note that $A'$ using $f$ is the same as $A$ using $f'$.

**Theorem 4.1.** *Let $f$ be a random function from $(b+c)$-bit blocks to $b$-bit blocks. Let $A$ be an oracle algorithm that performs at most $n$ distinct oracle queries. Let $\epsilon$ be the $A$-distance between $f'$ and uniform. Let $\delta$ be the $A'$-distance between $f$ and uniform. Then $\delta \geq \epsilon - \binom{n}{2} 2^{-b}$.*

Thus $f'$ is unpredictable if $f$ is unpredictable. For example, if $A$ breaks $\mathrm{surf}_k{}'$ with probability over $2^{-96}$, performing fewer than $2^{80}$ oracle queries, then $A'$ breaks $\mathrm{surf}_k$ with probability over $2^{-97}$. Here $b = 256$.

*Proof.* Let $g$ be a uniform random function from $(b+c)$-bit blocks to $b$-bit blocks. Then $\delta$ is the $A$-distance between $f'$ and $g'$, since $\Pr[A'(f) = 1] = \Pr[A(f') = 1]$ and $\Pr[A'(g) = 1] = \Pr[A(g') = 1]$. By Theorem 5.3, the $A$-distance between $g'$ and uniform is at most $\binom{n}{2} 2^{-b}$. ∎

**Generalizations.** The crucial property of $f^+$ is that it has **computationally uniform differences**: $f^+(p) - f^+(d)$ appears to be uniform for any fixed distinct inputs $p, d$. See Theorem 5.1. In particular $f^+(p)$ is almost never equal to $f^+(d)$. Unless the attacker stumbles across $p$ and $d$ with $f^+(p) = f^+(d)$, all his inputs to $f(\underline{0}, \cdot)$ will be distinct, so the outputs will appear to be independent.

The same function $f^+$ was used for authentication in [5]. The crucial property is again that $f^+$ has computationally uniform differences. See [24, Theorem 7].

There are alternatives to $f^+$ that are very fast for long messages and *provably* have almost uniform differences. See, for example, [24] and [28].

**Other constructions.** The usual fixed-length-input chaining MAC, which maps $p_1, p_2, p_3$ to $f(f(f(p_1) + p_2) + p_3)$, is unpredictable when $f$ is an unpredictable random function on $b$-bit blocks. This was proven by Bellare, Kilian, and Rogaway in [7].

Another available construction is cascading, proven unpredictable for fixed-length inputs by Bellare, Canetti, and Krawczyk in [3]. Cascading, unlike chaining and protected counter sums, demands a low-entropy random function.

Counter sums have several advantages over chaining and cascading, as noted in [5]. First, it is somewhat tricky to modify chaining and cascading to handle variable-length inputs. Second, chaining and cascading appear to lose security for long messages; counter sums do not. Third, chaining and cascading are inherently serial; counter sums can be evaluated quickly for long messages by a parallel machine.

## 5. The uniform case

The proof of Theorem 4.1 reduces to Theorem 5.3, proven below.

In this section "message" means "sequence of fewer than $2^c$ $b$-bit blocks."

**Theorem 5.1.** *Let $g$ be a uniform random function from $(b+c)$-bit blocks to $b$-bit blocks. Let $p$ and $d$ be distinct messages. Then $g^+(p) - g^+(d)$ is uniform.*

*Proof.* Say $p = (p_1, p_2, \ldots, p_k)$ and $d = (d_1, d_2, \ldots, d_l)$. The point is that, among $g(\underline{1}, p_1), \ldots, g(\underline{k}, p_k), g(\underline{1}, d_1), \ldots, g(\underline{l}, d_l)$, there is at least one term that is independent of all the rest. If $k > l$ then $g(\underline{k}, p_k)$ is it. If $k < l$ then $g(\underline{l}, d_l)$ is it. If $k = l$ and $p_i \neq d_i$ then $g(\underline{i}, d_i)$ is it.

That term is uniform. Thus the difference between $g(\underline{1}, p_1) + g(\underline{2}, p_2) + \cdots + g(\underline{k}, p_k)$ and $g(\underline{1}, d_1) + g(\underline{2}, d_2) + \cdots + g(\underline{l}, d_l)$ is also uniform. □

**Theorem 5.2.** *Let $g$ be a uniform random function from $(b+c)$-bit blocks to $b$-bit blocks. Let $q_1, \ldots, q_m$ be distinct messages, and let $r_1, \ldots, r_m$ be $b$-bit blocks. Then $g'(q_j) = r_j$ for all $j$ with probability at least $(1 - \binom{m}{2} 2^{-b}) 2^{-mb}$.*

*Proof.* By Theorem 5.1, $g^+(q_i) = g^+(q_j)$ with probability $2^{-b}$ for fixed $i \neq j$. Thus $g^+(q_1), \ldots, g^+(q_m)$ are distinct with probability at least $1 - \binom{m}{2} 2^{-b}$. Assume that in fact they are distinct; then $g(\underline{0}, g^+(q_1)), \ldots, g(\underline{0}, g^+(q_m))$ are uniform and independent, so they equal $r_1, \ldots, r_m$ with conditional probability $2^{-mb}$. □

**Theorem 5.3.** *Let $g$ be a uniform random function from $(b+c)$-bit blocks to $b$-bit blocks. Let $A$ be an oracle algorithm that performs at most $n$ distinct oracle queries. Then the $A$-distance between $g'$ and uniform is at most $\binom{n}{2} 2^{-b}$.*

*Proof.* Define $\epsilon = \binom{n}{2} 2^{-b}$. By Theorem 5.2, $g'$ is within $\epsilon$ of uniform on every set of size $m \leq n$, since $\binom{m}{2} 2^{-b} \leq \epsilon$. Apply Theorem 3.1. □

**Notes.** [7, Lemma 3.1], which is analogous to Theorem 5.3, is proven in [7] with several pages of analysis of various conditional probabilities. It would be easier to use the "end-to-end" approach illustrated here, combining Theorem 3.1 with a bound similar to Theorem 5.2.

## 6. An attack

The bound $\binom{n}{2} 2^{-b}$ in Theorem 5.3 is almost optimal whenever it is smaller than, say, 1%.

Indeed, fix $n \leq 2^b$, and fix $n$ distinct $b$-bit blocks $q_1, \ldots, q_n$. Consider the following algorithm $A$ using an oracle $h$: feed $q_1, \ldots, q_n$ to $h$; print 1 if the results $h(q_1), \ldots, h(q_n)$ are all distinct; otherwise print 0.

If $h$ is an oracle for a uniform random function then the results $h(q_k)$ are distinct with probability $p$, where $p = \prod_{0 \leq k < n} (1 - k 2^{-b})$.

If, on the other hand, $h$ is an oracle for $g'$, where $g$ is a uniform random function, then the results $h(q_k) = g(\underline{0}, g(\underline{1}, q_k))$ are distinct with probability $p^2$.

The $A$-distance between $g'$ and uniform is thus $p - p^2$. But $1 - x \leq p \leq 1 - x + x^2/2$ where $x = \binom{n}{2} 2^{-b}$; hence $p - p^2 = p(1-p) \geq (1-x)(x - x^2/2) = x(1-x)(1-x/2)$. Compare this lower bound to the upper bound of $x$ from Theorem 5.3. If $x < 1/100$ then the two bounds are almost exactly the same.

## Appendix. Basic probability theory

Probability theory considers a set Pr of possible universes. Pr is a probability space, i.e., a measure space of total measure 1. For an introduction to measure theory see, e.g., [25].

An **event** is a measurable subset of Pr. The measure of an event $E$ is called the **probability of** $E$, written $\Pr[E]$. For example, flip a fair coin. Let $E$ be the event that the coin comes up heads, i.e., the set of universes in which the coin comes up heads. Fairness means that the measure of $E$—the probability that the coin comes up heads—is $1/2$.

Let $E$ and $C$ be events, with the probability of $C$ nonzero. The **conditional probability of** $E$ **given** $C$ is the probability of $E \cap C$ divided by the probability of $C$.

**Random variables.** Let $X$ be a measurable space. A **random element of** $X$ is a measurable function from Pr to $X$. If the elements of $X$ are "objects" then "random element of $X$" is abbreviated as "random object."

Let $v$ be a random element of $X$. The **distribution of** $v$ is the measure on $X$ induced by $v$. Under this measure, $X$ is a probability space. For example, the result of a fair coin flip is a random element of the set $\{\text{heads}, \text{tails}\}$, with distribution $1/2$ heads, $1/2$ tails.

A random element of $X$ can be used in a formula as if it were an element of $X$. If $v$ is a random element of $X$, and $\varphi$ is a measurable function from $X$ to $Y$, then $\varphi(v)$—i.e., the composition of $\varphi$ and $v$—is a random element of $Y$.

Let $v_1, v_2, \ldots, v_n$ be random elements of various sets. If the distribution of $(v_1, v_2, \ldots, v_n)$ is the product measure induced by the distribution of each $v_i$ then $v_1, v_2, \ldots, v_n$ are **independent**.

**Uniform random variables.** The **uniform distribution** on a finite set $X$ is the measure assigning value $1/\#X$ to each element of $X$. A random element of $X$ is **uniform** if its distribution is uniform. For example, a uniform random 4-bit string is a random 4-bit string that takes each possible value with probability $1/16$.

Let $S$ and $T$ be sets, with $T$ finite. Let $X$ be the set of functions from $S$ to $T$. The **uniform distribution** on $X$ is the product measure induced by the uniform distribution on $T$. (For finite $X$ this is the same as the uniform distribution defined above.) If $s_1, s_2, \ldots, s_n$ are distinct elements of $S$, and $t_1, t_2, \ldots, t_n$ are elements of $T$, then a uniform random function $g : S \to T$ satisfies $g(s_1) = t_1$, $g(s_2) = t_2$, $\ldots$, $g(s_n) = t_n$ with probability $1/\#T^n$.

## References


[1] —, *37th annual symposium on foundations of computer science*, Institute of Electrical and Electronics Engineers, New York, 1996.

[2] Mihir Bellare, Ran Canetti, Hugo Krawczyk, *Pseudorandom functions revisited: the cascade construction and its concrete security*, in [1], 514–523.

[3] Mihir Bellare, Ran Canetti, Hugo Krawczyk, *Pseudorandom functions revisited: the cascade construction and its concrete security*, draft available as [2], newer draft available as `http://www-cse.ucsd.edu/~mihir/papers/cascade.ps.gz`.

[4] Mihir Bellare, Roch Guérin, Phillip Rogaway, *XOR MACs: new methods for message authentication using finite pseudorandom functions*, in [12], 15–28.

[5] Mihir Bellare, Roch Guérin, Phillip Rogaway, *XOR MACs: new methods for message authentication using finite pseudorandom functions*, draft available as [4], newer draft available as `http://www-cse.ucsd.edu/~mihir/papers/xormacs.ps.gz`.

[6] Mihir Bellare, Joe Kilian, Phillip Rogaway, *The security of cipher block chaining*, in [13], 341–358.

[7] Mihir Bellare, Joe Kilian, Phillip Rogaway, *The security of the cipher block chaining message authentication code*, draft available as [6], newer draft available as `http://www-cse.ucsd.edu/~mihir/papers/cbc.ps.gz`.

[8] Daniel J. Bernstein, *SEOC: Strategic Exportable Online Cipher*, draft, censored.

[9] Daniel J. Bernstein, *SURF: Simple Unpredictable Random Function*, draft available from `http://pobox.com/~djb/papers/surf.dvi`.

[10] Joan Boyar, *Inferring sequences produced by a linear congruential generator missing low-order bits*, Journal of Cryptology **1** (1989), 177–184.

[11] J. Lawrence Carter, Mark N. Wegman, *Universal classes of hash functions*, Journal of Computer and System Sciences **18** (1979), 143–154.

[12] Don Coppersmith (editor), *Advances in cryptology—CRYPTO '95*, Lecture Notes in Computer Science 963, Springer-Verlag, Berlin, 1995.

[13] Yvo Desmedt (editor), *Advances in cryptology—CRYPTO '94*, Lecture Notes in Computer Science 839, Springer-Verlag, Berlin, 1994.

[14] Shimon Even, Oded Kariv (editors), *Automata, langauges and programming*, Lecture Notes in Computer Science 115, Springer-Verlag, Berlin, 1981.

[15] Oded Goldreich, Shafi Goldwasser, Silvio Micali, *How to construct random functions*, Journal of the ACM **33** (1986), 210–217.

[16] Irving J. Good, R. F. Churchhouse, *The Riemann hypothesis and pseudorandom features of the Möbius sequence*, Mathematics of Computation **22** (1968), 857–861.

[17] Donald E. Knuth, *The art of computer programming, volume 2: seminumerical algorithms*, 2nd edition, Addison-Wesley, Reading, Massachusetts, 1981.

[18] Neal Koblitz (editor), *Advances in cryptology—CRYPTO '96*, Lecture Notes in Computer Science 1109, Springer-Verlag, Berlin, 1996.

[19] Michael Luby, Charles Rackoff, *How to construct pseudorandom permutations from pseudorandom functions*, SIAM Journal of Computing **17** (1988), 373–386.

[20] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, *Handbook of applied cryptography*, CRC Press, Boca Raton, Florida, 1996.

[21] Frederick C. Mish (editor), *Webster's ninth new collegiate dictionary*, Merriam-Webster, Springfield, Massachusetts, 1987.

[22] Harald Niederreiter, *The serial test for congruential pseudorandom numbers generated by inversions*, Mathematics of Computation **52** (1989), 135–144.

[23] Philip Rogaway, *Bucket hashing and its application to fast message authentication*, in [12], 29–42.

[24] Philip Rogaway, *Bucket hashing and its application to fast message authentication*, draft available as [23], newer draft available as `http://wwwcsif.cs.ucdavis.edu/~rogaway/papers/bucket.ps`.

[25] Halsey L. Royden, *Real analysis*, 3rd edition, Macmillan, New York, 1988.

[26] Adi Shamir, *On the generation of cryptographically strong pseudorandom sequences*, in [14], 544–550.

[27] Victor Shoup, *On fast and provably secure message authentication based on universal hashing*, in [18], 313–328.

[28] Victor Shoup, *On fast and provably secure message authentication based on universal hashing*, draft available as [27], newer draft available as `http://www.cs.wisc.edu/~shoup/papers/macs.ps.Z`.

[29] Alan M. Turing, *Computing machinery and intelligence*, MIND **59** (1950), 433–460.

Department of Mathematics, Statistics, and Computer Science, The University of Illinois at Chicago, Chicago, IL 60607–7045

*E-mail address*: `djb@pobox.com`