Journal of Network and Computer Applications 45 (2014) 121–133

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

# Secure and efficient random functions with variable-length output

Yan Zhu [a,*], Di Ma [b], Changjun Hu [a], Gail-Joon Ahn [c], Hongxin Hu [d]

[a] University of Science and Technology Beijing, Beijing 100083, China
[b] University of Michigan-Dearborn, Dearborn, MI 48128, United States
[c] Arizona State University, Tempe, AZ 85287, United States
[d] Clemson University, Clemson, SC 29634, United States

ARTICLE INFO

*Article history:*
Received 23 April 2013
Received in revised form
24 June 2014
Accepted 26 July 2014
Available online 5 August 2014

*Keywords:*
Algorithm
Randomness
Variable length
Random function
Hidden number problem

ABSTRACT

Many random functions, like Hash, MAC, PRG, have been used in various network applications for different security choices. However, they are either fast but insecure or cryptographic secure but slow. To integrate them together, in this paper we present a new family of square random functions, including SqHash, SqMAC and SqPRG, based on a specially truncated function (MSB or LSB), as well as circular convolution with carry bits. Provable security is provided by the privacy property in hidden number problem and Hard-core unpredication of one-way function. The experiment results show that these schemes have better performance under different input and output lengths. We also perform four types of statistical tests for randomness. The experiments indicate that our construction has good average-case randomness than SHA-2 and original Square algorithm.



## 1. Introduction

Random numbers are used widely in computers and networks for a variety of purposes. For example, a large number of scientific experiments and simulations need good random numbers to satisfy various requirements, such as creating randomized connections, choosing parameter values randomly, and injecting noise into network simulations (Shin and Weiss, 2010; Wu et al., 2009). The random number generators are also extremely useful in the game programming, in which randomness makes less deterministic for all players and therefore harder to defeat under simple patterns and more replayable (Peretz et al., 2011). Moreover, computer security requires some levels of encryption to be applied to various kinds of data, including secure Web transactions (Jalal and King, 2011), secure transmission technology (EDGE, Bluetooth, WiFi, WiMAX, etc.) (Pering and Want, 2012), or SSH (Gont and Bellovin, 2012). In addition, the random numbers are widely used in various applications, e.g., virus feature match, data comparison, and message lookup, so that a random number generator has already been a necessary function and module in the modern computer systems.

In general, cryptographic primitives, such as Hash function, message authentication code (MAC), and pseudorandom generator (PRG), have been widely used as the cornerstones of various random number generators (Lin et al., 2011; Huang et al., 2013). These functions are collectively called *random function* because a common feature among them is that their outputs are pseudorandom. In the cryptography, a random function is essentially a one-way function, which is easy to compute on any inputs, but hard to invert when given an output value. Typically, an efficient random function should satisfy some desirable properties, such as *randomness*, *security*, *efficiency*, to meet the requirements of various applications.

*Motivation*: Smart mobile devices (iPad, iPhone, Android, Blackberry) have experimented exponential growth over the last several years and there are currently around 1.2 billion users worldwide.[1] The explosive growth of wireless systems coupled with the proliferation of laptop and palmtop computers indicates a bright future for wireless networks. Many new applications, including wireless sensor networks, smart homes and appliances, and remote tele-medicine, are emerging from research ideas to concrete systems with the help of wireless networks.

Unfortunately, a wireless network is susceptible to security attacks because of its openness of transmission media. Mobile network security has become a major concern to mobile clients throughout the world. In order to solve this issue, more and more cryptographic techniques have been used to enhance the security of these mobile devices, such as signature, encryption, and identification. Especially, some cryptographic random functions, like Hash, MAC, PRF, have been widely employed in a variety of applications and services (Shamir, 2008), including warehouse inventory control,

* Corresponding author.Tel.: +86 13121328791.
*E-mail address:* zhuyan@ustb.edu.cn (Y. Zhu).

[1] http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats.

http://dx.doi.org/10.1016/j.jnca.2014.07.033
1084-8045/

public transportation passes, anti-counterfeiting tags for medicines, and secure identification.

In general, these cryptographic primitives are computed frequently on inputs which are often thousands of bytes long. Moreover, computing and verifying primitives are typically done in software, and may be done on relatively weak platforms with limited energy, such as, wireless sensor, bluetooth, and wearable computing. Additionally, the computations must be done in real time. Therefore, developing techniques for optimizing these cryptographic primitives while retaining an appropriate level of security is crucial to improve the performance of mobile devices.

More importantly, there exist too many algorithms (or standards) that can be used to implement these primitives. For example, general hash algorithms have FNV (Fowler et al., 1988), RIPEMD-160, MD5, SHA-1 (Sasaki and Aoki, 2009), SHA-2, HMAC (Bellare et al., 1996, 2000), Lookup3, Spooky, JSHash (Jenkins, 2009), and Murmur (Appleby, 2011). This often leads to a large number of different algorithms stored in the system, a waste of limited storage space, and even the confusion. Therefore, it is necessary to provide a common method to unify a variety of primitives into a family of cryptographic functions (Etzel et al., 1999) with variable-length output. So that we can use a common core algorithm to construct various cryptographic functions. In addition, special hardware or optimized software based on such a core algorithm can be designed to further improve the system performance. In brief, several goals desired in this paper have been found to be of great practical value, such as:

- To construct a family of random functions, including Hash, MAC, PRG, based on a common and light weight core algorithm.
- To implement a random function with variable-length output according to different requirements in various applications.
- To provide several significant properties, such as easy to understand and implement, provable security, parallelizability, and high performance of software and hardware.

Fortunately, a newly developed cryptographic technique, *hidden number problem* (HNP) on the lattice theory, provides us with a powerful tool for constructing such a family of random functions. For instance, it is well-known that the Square (Blum Blum Shub) hash algorithm, is provably secure, based on the difficulty of the quadratic residue problem (QRP). But only (loglog $N$) lower-order bits (Blum et al., 1986 (is considered as random in the output of QRP modulo a composite $N$, e.g., for a 1024-bit $N$, we can get a 10-bit random output. Especially, a recent survey from HNP showed that (log $N$)/3 bits still remains random (Boneh and Venkatesan, 1997; Kiltz, 2001). This means that we can get about 341-bit random output. Therefore, HNP should be an effective optimization technique to develop an efficient, provably secure, variable-length random function.

*Contribution*: In this paper, we focus on the construction of a family of random functions with a common, secure and efficient core algorithm. We first present a new hash scheme (called SqHash) based on a specially truncated function (most significant bits, MSB) of Square Hash function. This scheme can provide additional security for the inputted secret based on the assumption of hidden number problem. We also improve the performance of SqHash by using "circular convolution" which makes variable-length output possible. Furthermore, we present a new MAC scheme (called SqMAC) and a new PRG scheme (called SqPRG) based on the same core algorithm. We also prove that the security of these constructions based on the privacy property in the hidden number problem and the Hard-core unpredication of one-way function. Our experiment results show that these schemes have better performance under different input and output lengths. We also perform 4 types of statistical tests for randomness. The experiments indicate that our construction has good average-case randomness than SHA-2 and the original Square algorithm. The proposed schemes as well as their respective performance and security parameters are summarized in Table 1.

The remainder of this paper is organized as follows. In Sections 2 and 3, we review some preliminary background. In Sections 4–6, we present the new Hash, MAC and PRG constructions. In Sections 7–9, we analyze the security and performance features of our schemes and the improved schemes. Section 10 concludes the paper.

## 2. Related work

Hash function is one of the most basic forms of random function. It is related to (and often confused with) checksum, fingerprint, randomization function, and so on. Generally, hash functions can be divided into two main categories: non-cryptographic hash functions and cryptographic hash functions. The former neglects cryptographic security, so it is not cryptographically strong, but it offers these benefits: (1) it is extremely simple; (2) it is performed on bitwise and bit shift operations; and (3) it executes quickly on resource-limited processors. In Table 2 we show some non-cryptographic hash functions, such as, Spooky, FNV, Lookup3, and Murmur (Jenkins, 2009; Fowler et al., 1988; Appleby, 2011). However, this high performance makes it more feasible for a computer to find hash values (and thus collisions) by brute-force.

A cryptographic hash function is a cryptographic algorithm which is able to resist all types of attack. As a minimum, it must have the following properties: preimage resistance, second-preimage resistance, and collision resistance. Some existing schemes are designed on a mathematical problem and thus their

**Table 1**
Summary of our proposed schemes.

| Name | Item | Description |
|---|---|---|
| SqHash | Equation | $MSB_k((m\|\|IV)*'_c(m\|\|IV))$ |
| | Performance | $O(n*k)$ |
| | Security | Preimage resistance and collision resistance |
| SqMAC | Equation | $MSB_k((m+K)*'_c(m+K))$ |
| | Performance | $O(n*k)$ |
| | Security | Secret-key privacy on hidden number problem |
| SqPRG | Equation | $\widetilde{LSB}^u_{2l}((x+i+l)*'_c(x+i+l))$ |
| | Performance | $O(n*2l)$ |
| | Security | Pseudorandom on hard-core unpredictability |

**Table 2**
Hash function collection.

| Type | Hash fct. | Invented by | Ref. |
|---|---|---|---|
| Non-crypt. | Lookup3 | Bob Jenkins | Jenkins (2009) |
| | Spooky | Bob Jenkins | Jenkins (2009) |
| Hash | FNV | Fowler, Noll, Vo | Fowler et al. (1988) |
| | Murmur | Austin Appleby | Appleby (2011) |
| Crypt. | MD5 | Ronald Rivest | Sasaki and Aoki (2009) |
| | SHA | NSA | Sasaki and Aoki (2009) |
| Hash | FSB | Augot, Finiasz, etc | Saarinen (2007) |
| MAC | CBC-MAC | FIPS | Bellare et al. (2000) |
| | HMAC | Bellare, Canetti, etc | Bellare et al. (1996) |
| PRG | LCG[a] | | |
| | LFSR[b] | | |
| | BBS | Blum, Shub, etc | Blum et al. (1986) |

[a] Linear congruential generator.
[b] Linear feedback shift register.