Security of VSH in the Real World

Version: March 16, 2006 (b)

Markku-Juhani O. Saarinen

Information Security Group
Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK.
m.saarinen@rhul.ac.uk

Abstract. In this brief paper we offer commentary on the resistance of VSH, *very smooth hash*, against some standard cryptanalytic attacks. Although the authors of VSH claim only collision resistance, we show why one must be very careful when using VSH in cryptographic engineering, where additional security properties are often required.

1 Introduction

Many existing cryptographic hash functions were originally designed to be *message digests* for use in digital signature schemes. However, they are also often used as building blocks for other cryptographic primitives, such as pseudorandom number generators (PRNGs), message authentication codes, password security schemes, and for deriving keying material in cryptographic protocols such as SSL, TLS, and IPSec.

These applications may use truncated versions of the hashes with an implicit assumption that the security of such a variant against attacks is directly proportional to the amount of entropy (bits) used from the hash result. An example of this is the HMAC$-n$ construction in IPSec. Some signature schemes also use truncated hashes. Hence we are driven to the following slightly nonstandard definition of security goals for a practically useable hash function:

1. **Preimage resistance**. For essentially all pre-specified outputs x, it is difficult to find a message m such that $H(m) = x$. The difficulty should be $O(2^n)$ when there are n pre-specified bits in x.
2. **2nd-preimage resistance**. Given a pre-specified message m_1, it is difficult to find another message m_2 so that $H(m_1) = H(m_2)$. The difficulty should be $O(2^n)$ when there are n pre-specified bits that match in the hashes.
3. **Collision resistance**. It should require $O(2^{n/2})$ effort to find any two messages m_1 and m_2 that produce a collision $H(m_1) = H(m_2)$ in n pre-specified bits in the hashes.

In addition to the above three usual goals, we state a fourth, more informal goal – **pseudorandomness**. In essence, we would like a PRNG, stream cipher, or other derived design that relies on a hash function to have at least $O(2^{n/2})$ security, as if it was secured with a "real" pseudorandom function.

Pseudorandomness implies that a hash has good statistical properties and resistance against a wide array of distinguishing attacks.

All of the mentioned desirable properties are difficult if not impossible to prove without nonstandard assumptions. We note that proofs based on assumptions are themselves assumptions, whether their origins are in the traditions of symmetric or asymmetric cryptanalysis.

Relevance of Collision Resistance

Some researchers tend to concentrate their efforts on showing that their hash functions that provide collision resistance, while ignoring other security properties. However, it is well known thar collision resistance does not imply preimage-resistance or other important hash function properties.

To illustrate this point, we present a classic counter-example. Consider an $n+1$-bit hash $H'(x)$ that has been constructed from an n - bit hash H as follows:

$$\text{If } |x| < n-1 \text{ then } H'(x) = x \mid\mid 1 \mid\mid 0\,0\cdots 0.$$
$$\text{If } |x| \geq n-1 \text{ then } H'(x) = \mathrm{H}(x) \mid\mid 1.$$

That is, if the message x is less than $n-1$ bits long, $H'(x)$ consists of the message itself, a single 1 bit and a padding of zero bits. If the message is $n-1$ bits or longer, the resulting hash consists of a (secure) hash of x, followed by a single 1 bit.

It is easy to show that H' is collision resistant if H is. It is also easy to see that H' is *not* pre-image resistant for a large proportion of hash outputs, and that a slightly truncated version is *not* collision resistant.

2 The VSH Algorithm

We describe the VSH algorithm in its most basic form, essentially as it appears in section 3 of [1]. [1]

Let $p_1 = 2, p_2 = 3, p_3 = 5, \ldots$ be the sequence of primes. Let n be a large RSA composite. Let k, the block length, be the largest integer such that $\prod_{i=1}^{k} p_i < n$. Let m be a be an l-bit message to be hashed, consisting of bits $m_1, m_2, \ldots, m_l$, and assume that $l < 2^k$. To compute the hash of m:

1. Let x_0 = 1.
2. Let $L = \lceil l/k \rceil$ the number of blocks. Let $m_i = 0$ for $l < i \leq Lk$ (padding).
3. Let $l = \sum_{i=1}^{k} l_i 2^{i-1}$ with $l_i \in \{0, 1\}$ be the binary representation of the message length l and define $m_{Lk+i} = l_i$ for $1 \leq i \leq k$.

[1] There have been many changes to VSH, most recently in early March 2006 when message length padding was changed to be performed *after* the message been hashed, rather than at the beginning. Such small changes have significant implications on the development of practical attacks. The attacks discussed in this paper apply only to this particular version of VSH; other attacks may be devised on other variants.

4. For $j = 0, 1, \ldots, L$ in succession compute

$$x_{j+1} = x_j^2 \prod_{i=1}^{k} p_i^{m_{(jk+i)}} \mod n.$$

5. Return x_{L+1}.

Selecting a 1024-bit n modulus has been suggested in the original paper, indicating 131-bit block size k.

2.1 Preimage resistance

VSH is multiplicative: If a, b, and z be three bit strings of equal length, where z consists only of zeros bits and the strings strings satisfy $x \wedge y = z$. It is easy to see that

$$H(z)H(a \vee b) \equiv H(a)H(b) \pmod{n},$$

As a result VSH succumbs to a time-memory trade-off attack.

To solve the n-bit preimage M of $C = H(z)^{-1}H(M) \pmod{n}$:

1. Tabulate $H(x \mid\mid 00\cdots0)$ for $0 \leq x < 2^{n/2}$.
2. Do table lookups for $H(00\cdots0 \mid\mid y)^{-1}C$ for $y = 0, 1, 2, \ldots$, looking for a match.

The algorithm terminates when $M = x \mid\mid y$, in other words before $y < 2^{n/2}$. A preimage attack on VSH therefore has $O(2^{n/2})$ complexity rather than $O(2^n)$ as expected.

Note that the final squarings proposed in section 3 of [1] under subtitle "short message inversion" do not protect against this attack.

This type of attack is extremely serious if VSH is used to secure passwords, a typical application for hash functions. Password cracking time is effectively square-rooted by this attack. Note that the complexity of attack does not depend on the modulus size n, but on the entropy of the password strings.

2.2 One-wayness (of the "Cubing" Variant)

In section 3.4 of the VSH specification, a variant that uses cubing instead of squaring in its compression function is proposed. Using the Jacobi symbol, the compression function

$$x_{j+1} = x_j^3 \prod_{i=1}^{k} p_i^{m_i} \mod n,$$

becomes

$$\left(\frac{x_{j+1}}{n}\right) = \left(\frac{x_j}{n}\right) \prod_{i=1}^{k} \left(\frac{p_i}{n}\right)^{m_i}.$$

We define a "binary" version of the Jacobi symbol:

$$j(c, n) = \frac{1}{2}\left(1 - \left(\frac{c}{n}\right)\right).$$

We now have a linear equation giving the parity of some message bits:

$$j(x_{j+1}, n) = j(x_j, n) + \sum_{i=1}^{k} j(p_i, n) m_i \pmod 2.$$

Note that the Jacobi symbol can be very efficiently computed and that $j(p_i, n)$ is essentially randomly 0 or 1 for each randomly generated composite n. If the same message has been hashed with k different moduli n, a system of k linear equations can be obtained, leading to possible disclosure of bits using Gaussian elimination.

The same attack applies to the standard squaring version as well, but it only leaks information about the message length. This was not the case for VSH versions 3.57 and before.

One-wayness is implied by the standard hash security requirement of pre-image resistance. If one obtains some information about some of the pre-image bits easily, one can find the rest faster in an exhaustive search.

Hash functions have been used in cryptographic protocols as building blocks for MACs. If a MAC (or a digital signature) is not post-encrypted in a communications protocol, this will lead to disclosure of information about plaintext message bits.

2.3 Example of Collision Search in a Truncated VSH

We will first discuss a well-known generic technique for turning a partial collision attack into a full collision attack.

Assume that there is a fast $O(1)$ mapping f that causes the hash result to be in some smaller subset of possible outputs: $H(f(x)) \in S$, where $|S| < 2^n$. Typically f would be chosen in such a way that certain hash result bits are forced to have the same constant value. In other words, f forces partial collisions. Note that f itself should not produce too many collisions, i.e. $x_1 \neq x_2$ usually means that $f(x_1) \neq f(x_2)$.

If such an f can be found, and it is fast, the complexity of finding full collisions becomes $O(\sqrt{|S|})$. Note that f does not need to be able to force the hash to S on each iteration, it is sufficient that it works with reasonable probability. The iteration in low-memory parallel collision search algorithm becomes $s_{i+1} = H(f(s_i))$ [2].

Attack on a simple variant. We will instantiate this attack on a VSH variant that only uses the least-significant 128 bits of the hash function result. For basic VSH (1024-bit n, k=131) the result of hashing a 128-bit message $m_1|m_2|\cdots|m_{128}$ is:

$$x = \Big(19\big(\prod_{i=1}^{128} p_i^{m_i}\big)^2 \bmod n\Big) \bmod 2^{128}.$$

The constant $19 = p_8$ is caused by length padding. It is easy to see that modular reduction by n occurs in this case with less than 50% probability if m is random (or randomized) and its Hamming weight behaves accordingly. This will have an impact of less than one bit on the the complexity of the overall attack.

The squaring will reverse the effect, since we shall state (without proof) that 0^2, 1^2, 2^2, ..., $(2^{128}-1)^2$ maps to roughly $2^{125.4}$ (16.6%) different values $\bmod 2^{128}$.

Let $l = 42$. For each of the 2^l bit strings r of length l we precompute and store r, indexed by

$$\Big(\prod_{i=1}^{l} p_i^{r_i}\Big)^{-2} \bmod 2^l.$$

In the f function on each iteration we compute the partial product $(\prod_{i=l+1}^{128} p_i^{m_i})^2 \bmod 2^l$ and use that to select message bits $m_1, m_2, \ldots, m_l$ using the lookup table. This will usually force the least significant l bits to a certain constant value on each iteration. Hence we have "forced" the result into a significantly smaller subset with $O(1)$ effort. There are additional technicalities to this attack, but its overall complexity is:

- Pre-computing the table offline: $O(2^{42})$ time and space.
- Finding collisions: $O(2^{(126-42)/2}) = O(2^{42})$ time.
- Total cost: roughly $O(2^{42})$, rather than $O(2^{64})$.

We acknowledge that this represents just *one* way of truncating VSH – using, say, the most significant bits of the result would be an even worse option. Many other truncated variants can be attacked using a different f function.

2.4 Other features of VSH

The authors of VSH do not explicitly note this, but the hash function result can be updated after small changes without computing the entire hash again. A "bit flip" in a message will always cause a predictable change in the message result (it becoming multiplied $\bmod n$ by certain power of a small prime or its inverse). This is due to the highly algebraic nature of the hash.

We note such a property may be useful in some applications where rapid update of the hash is required, but it is undesirable in many more as it facilitates adaptive attacks against many cryptographic protocols.

3 Conclusions

In our opinion VSH is a simple, elegant design that is based on a plausible complexity-theoretic assumption (NMSRVS: Nontrivial Modular Square Root of Very Smooth Numbers). However, it should not be considered a general-purpose hash function as usually understood in security engineering.

Collision resistance is apparently the only security goal of VSH (the authors discuss other goals but give no sufficiently convincing results). VSH produces a very long hash (typically 1024 bits). There are no indications that a truncated hash offers any significant level of security. This appears to rule out the applicability of VSH in digital signature schemes which produce signatures shorter than the VSH hash result.

A "political" standardisation consideration is that (by definition) VSH has a back-door in the secret factorization of n. In the past it has been difficult to popularise cryptographic techonologies that rely on trusted third parties.

References

1. S. Contini, A.K. Lenstra and R. Steinfeld. VSH, an efficient and provable collision resistant hash function. IACR e-print 2005/193, 2005.
2. P. van Oorschot and M. Wiener. Parallel collision search with cryptanalytic applications. Journal of Cryptology, 12 (1999), p 1-28.