Classification of Hash Functions Suitable for Real-life Systems

Yasumasa Hirai [*], Takashi Kurokawa[†], Shin'ichiro Matsuo[*] ,
Hidema Tanaka[†] and Akihiro Yamamura[†]

Abstract

Cryptographic hash functions have been widely studied and are used in many current systems. Though much research has been done on the security of hash functions, system designers cannot determine which hash function is most suitable for a particular system. The main reason for this is that the current security classification does not correspond very well to the security requirements of practical systems. This paper describes a new classification which is more suitable for designing real-life systems. This classification is the result of a new theoretical classification and a new qualitative classification. We show a mapping between each class and standard protocols. In addition, we show new requirements for four types of hash function for a future standard.

1 Introduction

A cryptographic hash function provides certain security properties and plays a crucial role in building security applications related to digital signatures, authentication, and message integrity. It is also used to construct pseudorandom number generators and practical instantiation of the random oracle. The most commonly used hash functions are MD5 and SHA-1, designed by Ronald Rivest and by the National Security Agency (NSA), respectively. These have become de facto standards and are widely used in many applications. Recently, successful attacks against hash functions have been demonstrated by numerous researchers. Among these, the attacks by Wang [74, 75, 76] have had a great impact on both theoretical and practical research on hash functions. Wang's attack finds a pair of two distinct messages with the same hash digest of SHA-0 and MD5. It takes only a few hours to find collisions of two distinct messages for MD5. On the other hand, according to Wang's latest announcement, the estimated cost to find a collision for SHA-1 is 2^{63} . These attacks unveil the vulnerability of some applications using existing hash functions, and the replacement of existing hash functions can be a severe burden for vendors, system designers, and users. In particular, the vulnerability of hash functions affects the public key infrastructure scheme and the replacement of weakened hash functions by new ones in PKI schemes or real-life protocols will be a major project. We note that Bellovin and Rescorla discuss how to deploy a new hash function without harming the existing real-life systems. Hash functions are chiefly used for authentication and data integrity, and so the security provided can be measured by the lifespan of the data. Traditionally, the security notion of hash functions is discussed from a theoretical standpoint. Since hash functions are usually fixed techniques and the security parameter is not fed, the security is not measured asymptotically to the security parameter. On the other hand, it is sometimes necessary to consider the trade-off between security and performance because of some constraints. In this paper, we clarify the usage of cryptographic hash functions and the security requirement in practical security systems. In particular, we survey the usage in practical protocols specified in RFC and so on. Looking into practical applications, we observe that the security requirements are not properly evaluated. Therefore, a reexamination of the security concept of hash functions in protocols is required before replacement hash functions can be created. Hence, we discuss how to widen the scope of cryptographic hash functions and give a new classification for the future use of cryptographic hash functions in practical use. We also discuss changing the concepts of hash functions and their mode of operations.

In Section 2, we discuss the traditional classifications of hash functions and recall typical usages of hash functions in cryptographic schemes. In Section3, we list several usages hash functions in real-life systems. We exemplify that hash functions are inevitable in numerous information communication systems even though the strong security properties of hash functions are not necessarily required in some cases. In Section 4, we argue the several reasons for traditional classification is not suitable for the current usages in real-life systems. Then we shall discuss how to classify hash-related techniques and how to evaluate its security properties in Section 5. We exclude message au-

[*]NTT DATA Corporation, 1-21-2, Shinkawa, Chuo-ku, Tokyo 104-0033

[†]National Institute of Information and Communications Technology (NICT), 4-2-1, Nukui-Kitamachi, Koganei, Tokyo, 184-8795

$$\begin{tikzcd}
& \text{Cryptographic hash function} \arrow[dl] \arrow[dr] & & \\
\text{MAC} & & \text{MDC} \arrow[dl] \arrow[d] \arrow[dr] & \\
& \text{OWHF} & \text{UOWHF} & \text{CRHF}
\end{tikzcd}$$

Figure 1: Cryptographic hash functions

thentication codes (MAC) from our discussion because our goal is to discuss successor of the traditional collision resistant hash functions such as SHA-1. Then our proposal on classification will be given in Section 6 in which we emphasize four types of hash functions. The classification is made based on theoretical assurance, availability and performance. In Section 7, we consider additional requirements for hash functions.

2 Existing Cryptographical Classification

Hash functions are traditionally classified into two essentially distinct categories: non-cryptographic hash functions and cryptographic hash functions. Cryptographic hash functions are classified into message authenticated codes (MAC), one-way hash functions (OWHF), collision-resistant hash functions (CRHF), and universal one-way hash functions (UOWHF) (Figure 1). While we do not discuss any technical definition of these techniques, we consider the current usage of them in industrial scenes. Non-cryptographic hash functions have been chiefly used to search database systems and have had nothing to do with security systems. Cryptographic hash functions are almost always required to be collision resistant, and most of the instantiations in practical security systems are restricted to a few hash algorithms of the SHA-family or the MD4-family. Cryptographic hash functions are constructed using compression functions and composition methods.

Non-cryptographic hash functions may also require security to some extent. Of course, these need not be as secure as cryptographic hash functions. Here, we explain the necessity of reexamining such hash functions.

2.1 Security of cryptographic hash functions

Suppose $h : \{0,1\}^* \to \{0,1\}^n$ is a hash function. Cryptographic hash functions should have the following properties.

Collision resistant : The computational cost to find the input pair x and x' which holds $h(x) = h(x')$ must not be smaller than $2^{\frac{n}{2}}$, which is estimated with respect to the birthday paradox.

Pre-image resistant : The computational cost to find the input x, where $h(x) = y$ must not be smaller than 2^n.

Second pre-image resistant : The computational cost to find the input $x'(\neq x)$, where $h(x) = y$ and $h(x') = y$ must not be smaller than 2^n. It has been shown in [39] that the necessary number of algorithm step may be less than 2^n if immense memory is allowed, that is, a kind of time-memory trade-off methods is used.

Universal one-way : Choose an element $x \in D$. When a key K is randomly chosen, it is hard for an adversary to find $y \in D$ $(x \neq y)$ such that $h_K(x) = h_K(y)$.

2.2 Message authentication code

The message authentication code (MAC) is a keyed hash function. There are two types of popular MAC: HMAC, which is constructed from two hash functions, and those based on a block cipher such as CBC-MAC or CMAC. The security property of MAC is unforgeability. A block-cipher-based MAC is easier to forge than HMAC for an adversary who gets the key. Security against side-channel attacks should also be considered. We do not discuss MACs in this paper because our goal is to discuss successor of the traditional collision resistant hash functions such as SHA-1 and much of the research done on MACs is well known.

2.3 Collision resistant hash functions

Most collision-resistant hash functions $h : \{0,1\}^* \to \{0,1\}^n$ are constructed through the iterated use of compression function $f : \{0,1\}^{n+m} \to \{0,1\}^n$. Then the security of such hash functions depends on the design of compression functions and the iteration methods.

2.3.1 Construction

The Merkle-Damgård construction is a well known solution for the question of how to iterate a compression function [20, 51]. Tree construction is another way, but most practical hash functions are based on Merkle-Damgård or its extensions.

Some research has examined the security relation between f and h, but has not led to a practical improvement in security. Recent results (e.g., Wang et al.) of attacks have show that pseudo-collisions or near-collisions are a real threat to the Merkle-Damgård construction. A solution to these potential problems is needed.

2.3.2 Compression function

Block cipher based: Block ciphers are used as compression functions because they are highly trusted functions that provide practical security and are easy to evaluate. PGV-style construction is widely used [58]. On the other hand, they must execute the key set-up function many times, so the performance is very slow.

Dedicated function: The MD4-family that has evolved from MD4 to SHA-224/-256/-384/-512 is a de facto standard. Many types of dedicated function have been chosen as international standards (e.g., SHA-1,2, RIPEMD-160, Whirlpool). Until 1998, the results of an attack on a hash function were known only for the reduced MD4 [17] and the near collision of [73]. In 1998, MD4 was completely broken by Dobbertin [23]. Until 2004, analysis confirmed the security properties of the MD5 and SHA-1 compression functions and the collision search for reduced MD5, SHA-0, and SHA-1. In 2004, MD5 was completely broken by Wang et al. [74] and SHA-1 now seems close to being broken [75]. Though practical performance has been achieved, most dedicated functions are now in a critical situation.

Arithmetic function: MASH-1/-2, which are based on modular arithmetic, were among the functions chosen for ISO/IEC 10118-4. The functions based on the knapsack problem have unresolved problems regarding security. Some, such as VSH [15], have provable collision resistance, but most are not suitable for practical use.

2.3.3 Recent works

In response to the SHA-1 crisis, research has shown that security can be improved with minimal changes made to the algorithm. SHA-1 IME [72] has an improved form of message preparation compared to that of SHA-1, and randomized hash [30] is a technique which makes it difficult to find a collision pair through a randomizing message. In addition, new dedicated hash functions have been proposed (FORK-256, DHA-256, etc). While most hash functions have a fixed output length, there have been some attempts to enable a variable length. For example, SHA-V is a variable length hash function based on SHA-1 [31].

2.4 Random Oracle methodology

In practical systems, random oracles are instantiated by a hash function. Such instantiation does not necessarily provide a level of security equal to provable security. We should not depend too much on random oracle models: several studies have shown that various cryptographic systems are secure in the random oracle model, but insecure for any concrete instantiation of the random oracle [13, 29, 18]. At present, given the success of Wang's attack, we should concentrate on techniques such as signature schemes and time stamps. There are several approaches that enable provable security with or without depending on random oracle models [10, 16].

2.4.1 Full Domain Hash Scheme

Here, the output size of a hash function is exactly the size of the (RSA) modulus. Implementing such a scheme requires a concrete method, such as MGF1, to extend the length of a hash function. In general, it is desirable to have consistency among the hash sizes and other security parameters such as the size of the modulus of the composite numbers used in factoring-based systems. It may be better to have hash functions that allow choice regarding the digest size.

2.4.2 Mask generation functions

In some provably secure cryptosystems, a mask generation function, such as MGF1 [88, 86], is used as a building block for instantiating random oracles. For example, RSASSA-PSS [88] was recently included in NIST FIPS 186-3 [84] and SP800-78 [85].

MGF1 is based on the ideas of Bellare and Rogaway [5] and [6]. It is viewed as a custom method to extend the length of a hash function. Such a construction has been adopted to avoid structured operations in standard hash functions. This may not be completely unrelated to the design criteria of the hash function.

2.5 UOWHF(Universal one-way hash function)

UOWHFs (introduced by Naor and Yung [54]) can be considered keyed hash functions that satisfy the following property: an adversary chooses a message x and then the function h_K is chosen randomly; it then becomes hard for the adversary to find y $(x \neq y)$ such that $h_K(x) = h_K(y)$.

Signature schemes constructed based on the hash-and-sign method are no more secure than a hash function against a collision-finding attack. The recent attacks by Wang against SHA-1 call these methods into question.

It is well known that a simple modification of the hash-and-sign paradigm may replace the collision-resistant hash with a UOWHF [54, 7].

On the other hand, the UOWHF has several disadvantages. So far, UOWHFs are inefficient compared to CRHFs; the key size grows logarithmically with the message size. However, a few practical techniques currently use UOWHFs; for example, ACE-Sign [68] is based on UOWHFs and has a not so tight reduction [47].

This poor efficiency and other disadvantages have slowed the development and deployment of UOWHFs. For example, no standard technique has been established by any international standards organization. This means it is hard to achieve interoperability and difficult to deploy UOWHFs.

3 Hash Functions in Information Systems

Generally, a hash function is used to securely provide services through systems. However, many system developers and engineers use security products in which a hash function is applied, yet they are not conscious of each hash function's security requirements. They therefore choose hash functions such as MD5 and SHA-1 without being aware of how an attack on the hash function can affect a system. In this section, we describe hash function requirements from a system developer's viewpoint. First, we describe the requirements during the design of systems. After that, we describe the current usage of hash function in systems. We then categorize hash function usage with regard to the requirements.

3.1 Requirements in system design

The system design requirements are as follows.

Security requirements : Systems have to fulfill the following security requirements to provide services securely.

- **Confidentiality :** Systems which store and use data, such as individual information, have to protect it from eavesdropping by malicious users.
- **Authentication and Certification :** In some systems, such as those used for content delivery services, content providers have to identify users for payment. When operators log onto systems to update software, the systems need to confirm the validity of each operator to prevent internal crimes.
- **Integrity :** Some systems which issue certificates, such as a driver's license, have to prevent falsification of certificates and guarantee their validity.

System development requirements :

- **Choosing an algorithm :** The design or algorithms used in a system are chosen based on requirements with respect to performance and operating environment. For example, a light-weight algorithm is needed for low-power devices. Thus, algorithms that can satisfy various performance and environmental requirements should be prepared.
- **Development-cost :** In almost all cases, the development cost is decided beforehand, and this includes the cost of security mechanisms. However, most customers are unaware of the additional costs of replacing a broken hash function with a new one.
- **Development period and system life cycle :** Most information systems have a life cycle which includes development, operation, and renewal phases. The basic design and algorithm are decided early in the development phase. Thus, when a hash function is broken during a later phase, replacement of the hash function will be planned immediately before the next renewal.

Requirements for each service :

- **Compliance (Long-term assurance) :** In financial systems and healthcare systems, many documents must be securely preserved for a long time. For example, in the United States, SEC17a-4 states that e-mail messages regarding the dealing of securities and so on must be stored for at least five years. The Sarbanes-Oxley (SOX) Act obliges corporations to store fiscal reports, audit reports, and all business transaction and conference related documents for at least five years [89]. Furthermore, the Health Insurance Portability and Accountability Act (HIPAA) requires the storage of medical information for at least seven years [83]. If, for example, we use time-stamp technology to ensure the integrity of documents, this sort of legislation means that a time-stamping system must provide long-term assurance, such as a guarantee of at least five years.
- **Enforcement of products :** The Certificate Authority (CA) and Time Stamping Authority (TSA) systems, for example, must be high-level security systems. Therefore, customers typically request that these systems use a Hardware Security Module (HSM) that

has been validated under the Cryptographic Module Validation Program (CMVP).

3.2 Usage of hash functions in systems

Many information systems use security protocols with hash functions, such as SHA-1. In this section, we list the protocols used with a hash function, and the impact of a broken hash function.

3.2.1 Protocols used with Hash Function

Certification : This category consists of certification protocols which guarantee the validity of signers, documents, issuers, etc.

Digital Signature : Digital signature is an encryption scheme for authenticating digital data. In general, a digital signature scheme consists of both message authentication codes compressed by a hash function and digital signature schemes encoded by public key encryption. Most digital signature schemes use a hash function to ensure the integrity of messages.

PKIX : PKIX provides an Internet X.509 Public Key Infrastructure Certificate and CRL profiles. PKIX uses a hash function to ensure the integrity of the certificates and CRL profiles.

Time Stamping : The Time Stamping protocol provides a time stamp token which guarantees the time at which data existed. The Time Stamping protocol uses a hash function to ensure the integrity of the time stamp token.

Authentication : This category consists of authentication protocols which confirm clients, etc.

Kerberos : Kerberos provides authentication, prevents eavesdropping, and ensures the integrity of data on the client-server model. Kerberos uses a hash function to calculate the hash value of the entered client password and this becomes the secret key of the client.

IEEE802.1X-EAP : IEEE802.1X-EAP provides authentication, active key generation, and key delivery on wireless networks. The EAP-FAST protocols in IEEE802.1X-EAP are based on the TLS handshake protocol, which uses a hash function.

APOP : APOP provides authentication which enciphers the password used for reception of an e-mail message. APOP uses a hash function (MD5) to prevent replay attacks and disclosure of a shared secret.

RADIUS is also a protocol that uses a hash function (e.g., SHA-1) to ensure the integrity of data and applications in information systems.

Secure communication : This category consists of the secure communication protocols which include key exchange, etc.

IPsec : IPsec provides secure communication at the network layer by encrypting and authenticating IP packets. Internet Key Exchange (IKE) protocols in IPsec use hash functions as pseudo-random functions. Moreover, hash functions are used to ensure the integrity of all protocol messages in protocols.

SSL/TLS : Secure Socket Layer (SSL) and Transport Layer Security (TLS) enables authentication and secure communication on the Internet. These protocols prevent eavesdropping, falsification, and message forgery. The handshake protocol in SSL uses a hash function to create a message authentication code.

SSH : SSH authenticates users, and provides a secure channel between a local and a remote computer. SSH uses hash functions to produce message authentication codes that ensure the integrity of data exchanged between the two computers.

Secure E-mail : This category consists of secure e-mail protocols which encrypt e-mail messages to protect privacy and verify the integrity of messages. We show them as follows.

S/MIME : S/MIME provides authentication, message integrity protection, and privacy protection of data. S/MIME uses a hash function in a digital signature scheme to ensure the authenticity and integrity of e-mail messages.

PGP : PGP provides e-mail encryption and authentication. PGP uses a hash function to ensure the integrity of e-mail messages.

Other protocols : Protocols in this category do not have precise security requirements regarding the hash function.

Packet sampling and filtering : Packet sampling protocols select a representative subset of packets, and packet filtering protocols remove packets that are not needed. Packet sampling protocols use hash functions as pseudo-random functions to select packets at random. Packet filtering protocols use hash functions to generate packet digests which are used to identify packets.

Database retrieval : Hash functions are applied to generate keys used to efficiently retrieve target data from large bodies of data stored in a database.

Software download : Software download systems use hash values as check sums to ensure the integrity of software. Users confirm whether the software has been altered.

Intrusion Detection System (IDS) and DomainKeys Identified Mail (DKIM) are also protocols that use hash functions. IDS uses a hash function for the system integrity verifier (SIV), and DKIM uses a digital signature scheme which includes hash functions to ensure the message integrity.

We show the classification of protocols which uses hash function in systems, and influence on the attack in Figure 2.

4 Gaps between Theoretical Classification and Current Usage

In this section, we first discuss gaps between the area covered by the present cryptographic classification and the current usage of hash functions. After that, we state the classification requirements suitable for both worlds.

4.1 Lack of quantitative security

In Section 2, we described the existing hash function classifications. These classifications are based on qualitative analysis. However, the security of real-life systems is considered from the perspective of a risk analysis method such as ISMS, ISO/IEC 15408, ISO/IEC 27001-2005, and so on. The results of such an analysis are quantitative.

For example, a digital signature used in time-stamping services must be valid for over seven years for HIPPA, and a digital signature for certificates must be valid for one to three years and remain resistant against possible adversaries.

On the other hand, a hash value used in only one session must be valid until the end of the session. If a hash value is used as a cookie in a Web-based service to prevent a man-in-the-middle attack, the system must prevent generation of a collision for the hash value before the end of the session. Typically, a session does not last long (i.e., one hour or less). For another example, a hash value used in an APOP protocol does not require long-term collision resistance. To protect such systems from being needlessly weakened, a weaker security class should be added and a guideline for such usage provided. The problem in this area is there are no security criteria for such usage. In most cases, a system engineer embeds hash functions into a system without doing a formal security analysis. This is because most engineers lack sufficient knowledge of hash functions, and there is no formal measurement of hash functions for such usage.

When we want to use the current classification to design a secure system, a system engineer cannot easily judge whether any particular hash function is suitable for a specified system. Most hash functions ensure theoretical security, such as collision resistance, second pre-image resistance, and so on, but a system engineer cannot reliably determine when each hash value becomes invalid in the system. Thus, we must consider the quantitative security aspect of hash functions and include this in the classification.

4.2 Additions to theoretical classification

In governmental cryptographic standard such as NESSIE and CRYPTREC, "provable security" is quite important nature for cryptographic algorithms. Government recommend provable secure digital signature schemes. This is because, digital signatures in e-governments must be secure for strong adversaries.

On the other hand, current qualitative criteria for hash functions is collision resistance, second pre-image resistance and one-wayness as explained in Section 2. Collision resistance is seen as an important characteristics, because it is thought as most strong notion among these three. These three notions are all rigorously defined. However, there are no provable secure constructions of them.

Current most secure digital signature schemes are proven in random oracle model. However, all standard hash function is not the random oracle. Consequently, this shows there is no secure digital signature in real system. The existing qualitative classification does not cover all hash function usage. Thus, we must consider the following modifications to theoretical classification.

Collision resistance is sufficient for most usages. This is helpful for authentication, key exchange, and other light weight usages. However, it is not sufficient for high-level security such as several systems in e-government. To cover such usage, a theoretical class which ensures a "real" provable secure digital signature is needed additional to collision resistance.

4.3 Requirements for new classification

For the reasons given above, we must define a new system of classification which covers everything from cryptographically strong classes to light-weight but practically secure classes.

The requirements that such a classification must meet are as follows.

- The classification must include a quantitative index, which covers from long-term security assurance to short-term security assurance.
- Theoretical classification must be redefined to cover the entire range from provable security to light and practical security.

Thus, a new classification system will be based on a two-dimensional matrix containing both qualitative index values and quantitative index values.

5 New Security Classification of Hash Functions

5.1 Cryptographical security

5.1.1 Redefine of security

The security of hash functions has been evaluated only in terms of collision resistance as estimated by the birthday paradox, as shown by Yuval in 1979 [80]. The attack criteria with respect to the reduced round type or the properties of pseudo-collisions and near collisions have not been studied enough compared to the differential and linear properties of block ciphers. It is necessary to redefine collision resistance. In particular, for an iterated hash function, the present collision resistance is clearly not sufficient with respect to the security property. Thus, collision resistance should be classified using “full collision”, “near collision” and “pseudo-collision” categories.

Full collision is the same as an existing collision and the level of security against it can be estimated with respect to the birthday paradox. This corresponds to a brute force search and can be applied to the evaluation of a compression function f.

Near collision is a collision within a few bit positions. How the hash value is truncated (e.g., truncation of the output from SHA-256 to a 160-bit hash value) will become an important property. How to estimate a near-collision is an open problem. In a certain situation, near collision resistent property is useful, for example, near collision is crucial in [78].

Pseudo collision is a collision where $h(x;IV) = h(x';IV')$ holds. The Merkle-Damgård construction potentially has this weak property. In particular, it is important to estimate the level of security against an attack using a multi-message block, such as Wang's SHA-1 attack. How to estimate a pseudo-collision is an open problem.

5.1.2 Requirement of new CRHF

New CRHFs should be developed by taking into account the following points.

Need to improve the compression function f
This will greatly affect the security and performance of a CRHF. Dedicated designs should be devised to achieve a theoretical security property such as provably secure. Arithmetical functions should improve performance.

Need to improve the construction h
The Merkle-Damgård construction has the property $h(x_1||x_2;IV) = h(x_2;h(x_1;IV))$. In an actual system, the demand for iterated hash functions is consequently large because of their good performance and ease of use. Therefore, the iterated construction should be used to find another construction which can solve the pseudo-collision problem or provide minimal security regarding collision resistance in a theoretical way.

Size of hash value
In addition to the demand for iterated hash functions, as stated above, from the viewpoint of hash size compatibility a decision on a standard 128-bit or 160-bit CRHF is desirable. For such use, security requirements are within a limited range. It is necessary to show that this limited security is enough for such usage.

5.1.3 Requirement for UWOHF

Naor and Yung's method [54] is not really practical because the length of keys for a UOWHF becomes longer in proportion to the message size. Bellare and Rogaway [7] provide a better solution to this problem. At present, the best methods provide signature techniques with a UOWHF where the key grows logarithmically with the message size. Among these, Shoup [69] gives the most efficient construction, which can be considered an extended Merkle-Damgård construction.

Theoretical assurance of the security of signature schemes has attracted increased attention since Wang's attacks. A UOWHF has some advantages compared to a CRHF. It is theoretically possible to construct a UOWHF using only a one-way function. This assumption is much weaker than the assumption needed for CRHFs. Most signature techniques depending on CRHFs are vulnerable to the recent attacks of Wang et al. However, signature techniques have been introduced that depend only on UOWHFs [54, 63] and Wang's attack is not a threat to signatures using UOWHFs. In addition, the security of UOWHFs is not compromised by the birthday paradox attack. Thus, the cost of a generic attack on a UOWHF is 2^n if the size of the hash digest is n. Note that the cost of a generic attack (a birthday paradox attack) on a CRHF is $2^{\frac{n}{2}}$. There is a trade-off between such generic security criteria and usability; UOWHFs need keys and the key length grows logarithmically with the message size, whereas a CRHF needs no key (Table. 1).

Gennaro, Gertner, Katz and Trevisan [28] claim that the construction by [54] is optimal among a black box construction of UOWHFs using one-way functions. It follows that there is no effective way to construct a UOWHF in a universal manner. Therefore, it is necessary to construct an effective UOWHF directly from its theoretical primitives. In fact, we do not know whether an effective UOWHF actually exists. A worthy goal is to show whether an effective UOWHF construction exists and to standardize it if it does.

5.1.4 Mode of operations

The mode of operation related to hash functions would be desirable in many ways; it provides additional properties for security systems based on hash functions. The ability to change the length of hash digests would be an attractive feature. An appropriate mode of operation may allow the hash digest size to be changed. This is desired for the FDH-based signature scheme. In a random oracle model, we use many different mutually independent random oracles. The mode of operation may allow such hash function variations. In some applications, we need a hash function whose digests belong to particular mathematical structures, such as groups of elliptical curves. In such a case, it is not easy to encode hash digests to an element in the target domain, so it will be beneficial to provide a mode of operation for encoding hash digests to a specific domain. It is sometimes critical to specify the object identifier (OID) of hash functions, which is known as the hash firewall [37]. It may be possible to adopt a mode of operation that takes care of this.

5.2 Quantitative classification from valid period

Next, we add a quantitative classification. We categorize the quantitative security of hash functions into three groups.

Long-term security This category contains certificates assuring that data has not been forged over long periods of time; e.g., over five years to comply with the SOX Act. A time-stamp token issued by a time-stamp authority is an example of this category. Other possible examples are signed contracts and the some kind of digital signatures which are widely transferred.

In such applications, the system must prevent forgery of time-stamp tokens over long time spans. Thus, the hash functions used must be secure against any adversary for the same period of time. Here, we assume that a hash function which will be secure over five years must be used for systems in this category. In addition, any desired security parameter (e.g., the search space) must exceed 2^{128} to account for the possibility of a birthday attack.

Medium-term security This category contains certificates which assure the data has not been forged and these certificates must remain reliable for two to three years. Public key certificates are a good example of this. The valid period of a public key certificate varies according to the security policy of the certificate authority, and this may be two or three years. Credentials for certain services realized through digital signatures or hash values are another application. The valid period of such credentials is limited to within one or two years, and is sometimes only one month.

In this kind of application, the collision resistance of a hash function is important, but the required security is weaker than long-term security. Moreover, such operations may be done through smart cards, so light and minimally secure hash functions are desirable. Here, we assume that a hash function which will be secure anywhere from 1 month to five years should be used for systems in this category. In addition, the desired security parameter (e.g., the search space) must be 2^{80} to cope with birthday attacks.

Short-term security This category contains hash values and digital signatures used for key exchange, to prevent replay attacks and man-in-the-middle attacks, or for temporary authentication. The valid period of such values varies according to the services; however, this is limited to the period of service use. This is typically one month at the longest.

In this kind of application, the collision resistance of a hash function is still important, but the required security is weaker than medium-term security, and a lighter and minimally secure hash function is desirable. Some standard protocols require a short hash length, which is limited by the packet size and performance requests. Such usage also falls into this category. We assume that hash functions which will be secure for up to one month can be used for systems in this category. In addition, the desired security parameter (e.g., search space) should be 2^{64} to cope with birthday attacks. A shorter security parameter may be acceptable depending on the security policy of

	CRHF	UOWHF
Key	No	Key size grows logarithmically with the message size
Adversary Goal	Find $x, y \in D$ $(x \neq y)$ s.t. $h(x) = h(y)$	Choose $x \in D$ Given $h_K \in H$ Find $y \in D$ $(x \neq y)$ s.t. $h_K(x) = h_K(y)$
Compression functions	Dedicated functions Block cipher based Arithmetic	Strongly universal$_2$ functions
Construction methods	Merkle-Damgård Tree	XOR linear XOR tree Shoup (extended Merkle-Damgård)
Standard	ISO 10118-3	No standard exists

Table 1: Comparison between CRHF and UOWHF

Category	Period	Security parameter
Long-term	over 5 years	2^{128}
Medium-term	1 month - 5 years	2^{80}
Short-term	under 1 month	2^{64}

Table 2: Comparison among three categories

each system.

These three categories are compared in Table 2.

6 Our Proposal

In this section, we propose a new classification, then map the various types of usage described in Section 3.2 with respect to this classification.

6.1 New classification

The new classification is based on a two-dimensional matrix with qualitative index values and quantitative index values. The qualitative index has two members as described in Section 5.1. These are the CRHF and the UOWHF.

The quantitative index has three members, described in Section 5.2. These are long-term security, medium-term security, and short-term security. Consequently, the resulting matrix contains six cells.

Next, we map current forms of usage (described in Section 3.2) into the matrix. Table 3 shows the result of this mapping.

Time-stamping realized through the use of a digital signature must assure the validity of time-stamping tokens, and this depends on the ability to find collisions. Thus, it must take into consideration the provable security and long security parameter, and it is mapped onto a UOWHF with long-term security. S/MIME and PGP are also used for signing e-mail messages. These signed messages must guarantee the senders identity and the integrity of the e-mail; in some cases, the e-mail might be equivalent to a contract. Security again depends on whether collisions can be found. These methods are mapped into the same column as time-stamping. Code signing is used to authenticate the software vendor for each software application. The prevention of signature forgery by finding a collision is critical to ensure the security of code signing. Software with digital signatures may be used for long-term purposes. Thus, code signing is also mapped into the same column.

Time-stamping realized through only a hash function, such as with a linking protocol, must be collision resistant. Long-term security must be guaranteed as described above. Thus, a CRHF with a long security parameter is needed for this kind of time-stamping. The hash value for a software download is used to check the integrity of a downloaded file. Although the purpose is only to check the integrity, this hash value may be used for a long time so the hash function for this usage must be a CRHF and provide long-term security.

Public key certificate, which is standardized by PKIX, assures the correctness of a key pair in a public key cryptosystem. PKI is the foundation of public key cryptography, so the forgery of a public key certificate is a critical

	CRHF	UOWHF
Long-term	Certification (Time-stamping by hash func.) Integrity check (Software Download)	Certification (Time-stamping by signature, Code Signing) Secure E-mail (S/MIME, PGP)
Medium-term	N/A	Certification (PKIX)
Short-term	Secure communication (IPSEC, SSL/TLS, SSH) Authentication (IEEE 802.1X-EAP, Kerberos, APOP, DKIM) Other usage (Packet sampling/filtering, Database Retrieval)	N/A

Table 3: Proposed classification and mapping of current usage

issue with regard to PKI. Thus, a hash function in the certificate must consider provable security, so it should be a UOWHF however, the valid period of a public key certificate is generally under five years, so the hash function for PKIX should provide medium-term security.

Many security protocols, including IPSEC, SSL/TLS, SSH, IEEE802.1X EAP, and Kerberos, use hash values for their security. In these protocols, the hash value must be secure for a short time. Thus, the hash functions for such usage should be CRHFs with a short-term security parameter. DKIM, which is used to block spam e-mail or phishing mail, uses digital signatures for sender authentication. In this usage, the valid period of the digital signature can be short, because only one verification of the digital signature is required when a receiver obtains the e-mail. Thus, the hash function for DKIM should also be in the left column of Table 3. Hash values for packet sampling/filtering and database retrieval require collision resistance for only a short period. Thus, the hash functions for this usage are in the same column.

In table 3, only four of the six cells are occupied. These are the four types of hash function that should be considered to establish an industrial standard.

6.2 Desired hash function features

As shown in Table 3, four types of hash function are required for real-life information systems.

Type1. UOWHF with a long security parameter: In this type, the hash function must have universal one-wayness and consider the provable security. The hash function is required to preserve universal one-wayness over a long term. Thus, the security parameter of this hash function must be over 2^{128} and be extendable.

Type2. CRHF with a long security parameter: In this type, the hash function must be collision resistant and preserve its collision resistance over a long term. Thus, the security parameter of this hash function must exceed 2^{128} and the security parameter must be extendable.

Type3. UOWHF with a medium security parameter: In this type, the hash function must have universal one-wayness and consider the provable security. The hash function must preserve universal one-wayness over the medium term. Thus, the security parameter of this hash function must exceed 2^{80}.

Type4. CRHF with a short security parameter: In this type, the hash function must be collision resistant, and preserve its collision resistance over the short term. Thus, the security parameter of this type of hash function may be about 2^{64}. A shorter security parameter may be chosen if security conditions permit

The above four types can cover almost all the existing usage of hash functions. Thus, they meet the requirements for hash functions suitable for real-life systems. We believe that the design of future standard hash functions should focus on these four types and that any standardization process should consider each type.

7 Discussions

Hash functions have to satisfy other requirements from a system designer's point of view. Two additional points should be considered when designing hash functions.

Compatibility with existing systems When a system engineer plans to replace a weak hash function in a system with a stronger one, he must consider the replacement's compatibility with the existing system's design to minimize the replacement cost. The main issue is usually the hash value length. When designing a system, the designer decides which data structures to use for communication messages, databases, and so on. Changing the hash value length can necessitate system re-design, re-coding, and re-testing. The affected parts include not only hash-related processing, but also non-cryptographic processing. Thus, designing strong hash functions with the same output length is preferable for most system designers. The inclusion of this consideration into the requirements of a future hash standard should be further discussed.

Implementation for embedded hardware The key devices in current security systems include tamper-resistant devices such as smart cards. Smart cards are used for user authentication, key management, signing documents, and so on.

For example, in general PKI, a key pair of a public key cryptosystem is securely stored in the smart card of each user, and a digital signature and ciphertext is calculated within the smart card. Thus, a hash function must be implemented into the smart card. Many smart cards implement MD5 or SHA-1, but there is no commercial smart card which implements SHA-256/384/512. The main reasons for this are the processing speed and the working memory size. Smart cards are widely used in current systems, and there is no good alternative for them at present, so replacing the hash functions used in today ̓s smart cards is a major concern regarding the hash function transition in information systems. Any future hash standard must include a family of hash functions which are easy to implement in smart cards.

References

[1] B. Aboba and P. Calhoun, "RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)," RFC3579.

[2] C. Adams, P. Cain, D. Ponkas and R.Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)," RFC3161.

[3] J. Arkko, V. Torvinen, G. Camarillo, A. Niemi and T. Haukka, "Security Mechanism Agreement for the Session Initiation Protocol (SIP)," RFC3329.

[4] M.Bellare and D.Micciancio, "A new paradigm for collision ? free hashing: Incrementarity at reduced cost," Advances in Cryptology, EUROCRYPT97, LNCS 1233,Springer-Verlag (1997), pp.163-192.

[5] M.Bellare and P.Rogaway, "Optimal Asymmetric Encryption," Advances in Cryptology, EUROCRYPT'94 LNCS 950, Springer-Verlag(1995), pp. 92-111.

[6] M.Bellare and P.Rogaway, "The Exact Security of Digital Signatures-How to Sign with RSA and Rabin," Advances in Cryptology, EUROCRYPT'96 LNCS 1070, Springer-Verlag (1996), pp. 399-416.

[7] M.Bellare, and P.Rogaway, "Collision-Resistant Hashing: Towards Making UOWHFs Practical," Advances in Cryptology - CRYPTO '97, LNCS 1294, Springer-Verlag (1997), pp.470-484.

[8] S. Bellovin and E. Rescorla, "Deploying a New Hash Algorithm," the NIST Hash Function Workshop, 2005.

[9] S. Blake-Wilson, M.Nystrom, D Hopwood, J. Mikkelsen and T. Wright, "Transport Layer Security (TLS) Extensions," RFC3546.

[10] A. Boldyreva and M. Fischlin, "Analysis of Random Oracle Instantiation Scenarios for OAEP and other Practical Schemes," Advances in Cryptology - CRYPTO 2005, LNCS 3621, Springer-Verlag (2005), pp. 412-429.

[11] N. Cam-Winget, D. McGrew, J. Salowey and H. Zhou, "The Flexible Authentication via Secure Tunneling Extensible Authentication Protocol Method (EAP-FAST)," draft-cam-winget-eap-fast-03.

[12] R. Canetti, "Towards Realizing Random Oracles: Hash Functions That Hide All Partial Information," Advances in Cryptology - CRYPTO '97, LNCS 1294, Springer-Verlag (1997), pp.455-469.

[13] R. Canetti, O. Goldreich, S. Halevi, "The random oracle methodology, revisied," In STOC'98. ACM, 1998.

[14] J.L.Carter and M.N.Wegman, “Universal classes of hash functions,” *JCSS* **18** (1979), pp.143-154.

[15] S.Contini, A.K.Lenstra and R.Steinfeld “VSH, an Efficient and Provable Collision Resistant Hash Function,” http://eprint.iacr.org/2005/193

[16] J. Coron, Y. Dodis, C. Malinaud, P. Puniya, “Merkele-Damgård Revised: How to Construct a Hash Function,” Advances in Cryptology - CRYPTO 2005, LNCS 3621, Springer-Verlag (2005), pp. 430-448.

[17] B. den Boer and A. Bosselaers, “An Attack on the Last Two Rounds of MD4,” Advances in Cryptology CRYPTO91, LNCS 576, Springer-Verlag (1992), pp. 194-203.

[18] Y. Dodis, R. Oliveria and K. Pietrzak, “On the Generic Insecurity of the Full Domain Hash,” Advances in Cryptology - CRYPTO 2005, LNCS 3621, Springer-Verlag (2005), pp. 449-466.

[19] I. Damgård, “Collision-Free Hash Functions and Public-Key Signature Schemes,” Advances in Cryptology - EUROCRYPT ’87 LNCS 304, Springer-Verlag (1988), pp.203-216

[20] I.Damgård, “A Design Principle for Hash Functions,” Advances in Cryptology - CRYPTO ’89, LNCS 435,Springer-Verlag(1990) pp.416-427

[21] T. Dierks and C.Allen, “The TLS Protocol,” RFC2246.

[22] S. Dusse, P. Hoffman, B. Rmsdell, L. Lundblade and L. Repka, “S/MIME Version 2 Message Specification,” RFC2311.

[23] H.Dobbertin, “Cryptanalysis of MD4,” J. Cryptology 11(4),1998 pp.253-271.

[24] S. Dusse, P. Hoffman, B. Rmsdell and J. Weinstein, “S/MIME Version 2 Certificate Handling,” RFC2312.

[25] M. Elkins, D. Del Torto, R. Levien and T. Rossler, “MIME Security with OpenPGP,” RFC3156.

[26] S. Farrell, T. Kause and T.Mononen, “Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP),” RFC4210.

[27] M.Gebhardt, G. Illies and W. Schindler, “A Note on the Practical Value of Single Hash Collisions for Special File Formats,” the 1st Cryptographic Hash Workshop, 2005.

[28] R.Gennaro, Y.Gertner, J.Katz and L.Trevisan, “Bounds on the efficiency of generic cryptographic constructions,” *SIAM J. Comp.* **35** (2005), pp.217-246.

[29] S. Goldwasser and Y. Tauman, “On the (In)security of the Fiat-Shamir Paradigm,” In Proceedings of the 44th Annual IEEE Symposium of Foundations of Computer Science (2003), pp. 102-114.

[30] S.Halevi and H.Krawczyk, “Randomized Hashing: Secure Digital Signatures without Collision Resistance,” http://www.ee.technion.ac.il/ hugo/rhash.pdf

[31] Y.S.Her and K.Sakurai, “Analysis and Design of SHA-V and RIPEMD-V with Variable Output-Length,” http://itslab.csce.kyushu-u.ac.jp/ ysher/14.pdf

[32] D.Hong, B.Preneel and S.Lee, “Higher order universal one-way hash functions,” Advances in Cryptology - ASIACRYPT 2004, LNCS 3329, Springer-Verlag (2004), pp.201-213.

[33] D.Hong, J.Sung, S. Hong and S. Lee, “How to Construct Universal One-Way Hash Functions of Order r,” Progress in Cryptology - INDOCRYPT 2005, LNCS 3797, Springer-Verlag (2005), pp.90-103.

[34] R. Housley, W. Polk, W. Ford and D. Solo, “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile,” RFC3280.

[35] C.Hsiao and L.Reyzin, “Finding collisions on a public road, or one-way hash functions,” Advances in Cryptology - CRYPTO 2004, LNCS 3152, Springer-Verlag (2004), pp.201-213.

[36] R.Impagliazzo and M.Naor, “Efficient cryptographic schemes provably as secure as subset sum,” J. Cryptology 9 (1996), pp.199-216.

[37] B. Kaliski, “On Hash Function Firewalls in Signature Schemes,” The Cryptographers’ Track at the RSA Conference 2002, LNCS 2271 , Springer-Verlag(2002), pp. 1-16.

[38] J.Kelsey and T.Kohno, “Herding Hash Functions and the Nostradams Attack,” Advances in Cryptology - Eurocrypt 2006, Springer-Verlag (2006).

[39] J. Kelsey and B. Schneier, “Second Preimages on n-Bit Hash Functions for Much Less than 2n Work,” Advances in Cryptology - Eurocrypt 2005, LNCS 3494, Springer-Verlag (2005), pp. 474-490.

[40] S. Kent and R. Atkinson, "IP Encapsulating Security Payload (ESP)," RFC2406.

[41] G.Kim and E.Spafford, "design and implementation of Tripwire: A file system integrity checker," In Proceedings of the 2nd ACM Conference on Computer and Communications Security, November 1994.

[42] G.Kim and E. Spafford, "Experiences with Tripwire: Using integrity checkers for intrusion detection. In Proceedings of Systems Administration, Networking, and Security III, 1994.

[43] G.Kim and E. Spafford, "Writing, supporting, and evalutaing tripwire: A publically available security tool.," In Proceedings of the Usenix UNIX Applications Development Symposium, 1994.

[44] J.H.Kim, D. R. Simon and P. Tetali, "Limits on the Efficiency of One-Way Permutation-Based Hash Functions," 40th Annual Symposium on Foundations of Computer Science (FOCS), (1999).

[45] J. Klensin, R. Catoe and P. Krumviede, "IMAP/POP AUTHorize Extension for Simple Challenge/Response," RFC2095.

[46] A.K. Lenstra and E.R. Verheul, "Selecting Cryptographic Key Sizes," Journal of Cryptology (2001), vol. 14, no. 4, pp. 255-293.

[47] The NESSIE book (draft version April 19 2004). Available at http://www.cryptonessie.org.

[48] P. Hoffman, "Algorithms for Internet Key Exchange version 1 (IKEv1)," RFC4109.

[49] W. Lee, D. Chang, S. Lee, S, Sung and M. Nandi "Construction of UOWHF: Two New Parallel Methods," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* **E88-A(1)** (2005), pp.49-58.

[50] W. Lee, D. Chang, S. Lee, S, Sung and M. Nandi, "PGV-style Block-Cipher-Based Hash Families and Black-Box Analysis," *IEICE transaction on Fundamentals*, **E88-A, no.1** (2005), pp.39-48.

[51] R. Merkle, "One Way Hash Functions and DES," Advances in Cryptology - CRYPTO '89, LNCS 435, Springer-Verlag(1990), pp.428-446.

[52] I.Mironov, "Hash Functions: From Merkle-Damgard to Shoup," Advances in Cryptology - EUROCRYPT 2001, LNCS 2045, Springer-Verlag(2001), pp.166-181.

[53] I.Mironov, "Collision-resistant no more: Hash-and-sign paradigm revisited," Public Key Cryptography - PKC 2006, LNCS 3958, Springer-Verlag(2006), pp.140-156.

[54] M.Naor and M.Yung, "Universal one-way hash functions and their cryptographic applications," Proc. 21st STOC (1989), pp.33-43.

[55] C. Neuman, S. Hartman and K. Raeburn, "The Kerberos Network Authentication Service (V5)," RFC4120.

[56] J. Peterson, "S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP)," RFC3853.

[57] W. Polk, R. Housley and L. Bassham, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC3279.

[58] B.Preneel, R.Govaerts and J.Vandewalle, "Hash Functions Based on Block Ciphers: A Synthetic Approach," Advances in Cryptology - CRYPTO '93, LNCS 773, Springer-Verlag(1993) pp.368-378.

[59] B. Ramsdell, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Certificate Handling," RFC3850.

[60] B. Ramsdell, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification," RFC3851.

[61] C. Rigney, W. Willats and P. Calhoun, "RADIUS Extensions," RFC2869.

[62] C. Rigney, S. Willens, A. Rubens and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)," RFC2865.

[63] J.Rompel, "One-way functions are necessary and sufficient for secure sugnatures," Proc. 22nd STOC (2000), pp.387-394.

[64] S. Santesson and R. Housley, "Internet X.509 Public Key Infrastructure Authority Information Access Certificate Revocation List (CRL) Extension," RFC4325.

[65] P.Sarkar, "Masking-based domain extenders for UOWHFs: bounds and constructions," IEEE Transactions on IT **51** (2005), pp.4299-4311.

[66] J. Schaad, "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)," RFC4211.

[67] J. Schaad, B. Kaliski and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC4055.

[68] T.Schweinberger and V.Shoup, "ACE: The advanced cryptographic engine,", August 14, 2000. Available at http://www.shoup.net.

[69] V.Shoup, "A composition theorem for universal one-way hash functions," Advances in Cryptology - EUROCRYPT 2000, LNCS 1807, Springer-Verlag(2000), pp.445-452.

[70] D. Simon, "Finding Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions?," Advances in Cryptology - EUROCRYPT '98, LNCS 1403, Springer-Verlag(1998), pp.334-345.

[71] R.Steinfeld,J. Pieprzyk and H. Wang, "Higher Order Universal One-Way Hash Functions from the Subset Sum Assumption," Public Key Cryptography - PKC 2006, LNCS 3958, Springer-Verlag(2006), pp.157-173.

[72] M.Szydlo and Y.L.Yin, "Collision-Resistant Usage of MD5 and SHA-1 Via Message Preprocessing," Topics in Cryptology - CT-RSA 2006, LNCS 3860, Springer-Verlag(2006), pp.99-114

[73] S. Vaudenay, "On the Need for Multipermutations: Cryptanalysis of MD4 and SAFER," Fast Software Encryption FSE95, LNCS 1008, Springer-Verlag(1995), pp. 286-297

[74] X.Wang, D.Feng, X. Lai, and H. Yu, "Collisions for Has Functions MD4, MD5, HAVAL-128 and RIPEMD," IACR Eprint archive 2004/199, Aug. 2004

[75] X. Wang, Y. L.Yin, and H. Yu, "Finding Collisions in the Full SHA-1," Advances in Cryptology - CRYPTO 2005, LNCS 3621, Springer-Verlag(2005), pp.17-36.

[76] X.Wang, A.Yao, and F.Yao, "Cryptanalysis on SHA-1," Proceedings of Cryptographic Hash Workshop,

[77] M.N.Wegman and J.L.Carter, "New hash functions and their use in authentication and set equality," *JCSS* **22** (1981), pp.265-279.

[78] A.Yamamura and H.Ishizuka, "Detecting errors and authentication in quantum key distribution," Information Security and Privacy (ACISP2001), LNCS 2119, Springer-Verlag(2001), pp.260-273.

[79] T. Ylonen and C. Lonvick, "The Secure Shell (SSH) Transport Layer Protocol," RFC4253.

[80] G. Yuval, "How to swindle Rabin," Cryptologia (3)(1979), pp.187-190.

[81] T. Zseby, M. Molina, N. Duffield, S. Niccolini and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection," draft-ietf-psamp-sample-tech-07.

[82] G. Zorn, D. Leifer, A. Rubens, J. Shriver, M. Holdrege, I. Goyre, "RADIUS Attributes for Tunnel Protocol Support," RFC2868.

[83] "Health Insurance Portability and Accountability Act,"
http://www.cms.hhs.gov/HIPAAGenInfo/

[84] National Institute of Standards and Technology, "Draft Federal Information Processing Standard (FIPS) 186-3 - Digital Signature Standard (DSS)," March 2006. Available at http://csrc.nist.gov/.

[85] National Institute of Standards and Technology, "NIST Special Publication 800-78: Cryptographic Algorithms and Key Sizes for Personal Identity Verification," April 2005. Available at http://csrc.nist.gov/.

[86] IEEE, "IEEE Std 1363-2000: Standard Specifications for Public-Key Cryptography," January 2000
.

[87] "Information technology – Security techniques – Time-stamping services –," ISO/IEC18014-2.

[88] RSA Laboratories, "PKCS#1 v2.1: RSA Cryptography Standard," June 2002. Available at http://www.rsasecurity.com/rsalabs/.

[89] "Sarbanes-Oxley Act of 2002,"
SOX:www.sec.gov/about/laws/soa2002.pdf

Protocols	Requirements	IETF RFC	Security				Development			Service	Influence
			Confidentiality	Authentication	Certification	Integrity	Performance	Cost-Effectiveness	Development Period		
Certification	Digital Signature	-			○	○	smart card		Long	DKIM: Short-term Document: Medium-term	Large: Revocation Resigning
	PKIX	X.509 Cert & CRL: RFC3280 Extensions: RFC4325								Long-term assurance	
	Time Stamp	RFC3161									
Authentication	Kerberos	RFC4120		○		○			Medium	Short-term	small
	IEEE802.1X-EAP	draft-cam-winget-eap-fast03.txt									
	APOP	RFC2195 RFC2095	○						Short	Short-term	Small
Secure Communication	IPsec	IKE: RFC4109 ESP: RFC2406		○	○	○			Medium	Short-term	small
	SSL/TSL	TLS Protocol: RFC2246 Extensions: RFC3546									
	SSH	RFC 4253									
Secure E-mail	S/MIME	Ver2: RFC2312, RFC2311 Ver3.1: RFC3851, RFC3850		○		○			Medium	Long-term	Large: Resigning
	PGP	RFC3156									
Others	Packet Sampling Filtering	draft-ietf-psamp-sample-tech-07.txt							Medium	Short-term	Small
	Database Retrieval	-							Medium	Medium-term	-
	Software Download	-				○			Short	Long-term	Medium: Recalculation of hash value

Figure 2: Requirements for Hash function in systems