Resilience against Brute Force and Rainbow Table Attacks using Strong ICMetrics Session Key Pairs

Ruhma Tahir, Huosheng Hu, Dongbing Gu,
Klaus McDonald-Maier

School of Computer Science and Electronic Engineering
University of Essex
Colchester, United Kingdom
rtahir@essex.ac.uk, hhu@essex.ac.uk,
dgu@essex.ac.uk, kdm@essex.ac.uk

Gareth Howells

School of Engineering and Digital Arts
University of Kent
Canterbury, United Kingdom
W.G.J.Howells@kent.ac.uk

Abstract—**Cryptography has become an essential for providing security in embedded system applications. The employed cryptographic primitives should provide strong protection such that the security of the system is not compromised at any point in the lifecycle of a secure operation. This particularly includes the secure generation and maintenance of cryptographic keys. In general this assumption is difficult to accomplish, since there are attacks that come under this umbrella ranging from brute force attacks on the key to capturing the node to extract the key. In this paper we investigate and analyze ICMetrics and its counterpart scheme referred to as the scheme for the generation of strong high entropy ICMetrics session key pairs. ICMetrics is a key technology that computes the secret key based on hardware/ software properties of a device, thereby providing resilience against node capture attacks, while high entropy key pair generation scheme is employed to strengthen the generated ICMetrics basis number, so as to safeguard the generated strong key pairs from brute force and rainbow table attacks.**

Keywords-Brute force attacks, rainbow table attacks, ICMetrics, cryptographic key.

I. INTRODUCTION

In today's world, all cryptographically secure applications require authentication of users/devices for secure communication to take place. The notion of proper and trustworthy authentication revolves around using appropriate keys of sufficient strength for carrying the secure operations. This mainly depends on the appropriate generation and protection of cryptographic keys, so that there is no threat to integrity, confidentiality and availability of the cryptosystem. Without appropriate generation and handling of keys, keys could be easily guessed, modified, or substituted by unauthorized personnel who could then intercept sensitive communications [10].

Cryptographic algorithms that are used to provide secure communication in embedded system applications depend on the use of stored encryption/decryption keys. These algorithms have the inherent disadvantage that the compromise of an embedded system device, can lead to key being revealed to the adversaries [5]. Integrated Circuit Metrics or ICMetrics [2-4] is an alternative to stored encryption/decryption keys that uses unique measurable properties and features of a hardware device to generate a value that can serve as a key for cryptographic operations for a device. However the generated ICMetrics key suffers from weaknesses of low entropy and short length making it easy to compromise for an attacker. Therefore for the ICMetrics key to be used for cryptographic operations, it must be strengthened, bringing an increase in entropy and length of key, so that it cannot be compromised at any point of a cryptographic communication [14].

The scheme for the generation of strong ICMetrics session key pairs is designed for the purpose of improving the entropy and length of the key so that the generated ICMetrics key pairs are safe against attacks on cryptographic keys. As mentioned above, the core challenge with the ICMetrics generated secret key is the entropy and length of the generated secret value. The ICMetrics key can be safeguarded from different attacks on keys by employing schemes to strengthen weak keys, by increasing the length and entropy of the key. A scheme for the generation of high entropy ICMetrics session key pairs is proposed in [11-12] that propose the generation of strong ICMetrics key pairs of sufficient length using SHA-2 based key derivation function [9], [15]. The scheme iterates through multiple rounds of the SHA-2 based hash function to stretch the secret value to the required length, thereby also generating a key with high entropy. This increase in length of the ICMetrics key safeguards the key from brute force attacks [6].

Hashing and key stretching provide a layer of security for the key generation however if the hashed key is compromised, it is still vulnerable to rainbow table attacks. The attacker can easily use previously calculated hash and corresponding plaintext to crack the hash. To defeat the rainbow table attack on strong ICMetrics session key pairs, the proposed scheme further incorporates the use of random session ID assigned to each participating device that helps defeat the generated high entropy ICMetrics keys from rainbow table attacks [7]. The session key pairs are temporarily generated for every session and remain valid precisely for one communication session.

The remainder of this paper is organized as follows; in section 2 we discuss the attacks inherent to cryptographic keys. In particular the brute force and rainbow table attacks on ICMetrics keys is discussed in section 3, which is the focus of this paper. Section 4 discusses the ICmetrics technology and its features. An overview of a strong ICMetrics session key pair generation scheme is given in Section 5 To fully understand the threat model, section 6 outlines how ICmetrics will safeguard against node capture attacks by taking two different scenarios of possible brute force and rainbow table attacks. The final section concludes our paper.

II. Attacks on Cryptographic Keys

Sensitive embedded system applications require appropriate key generation mechanisms to be used for the generation of strong cryptographic keys. These strong cryptographic keys must possess properties of sufficient entropy and length so that they cannot be easily cracked by an adversary. Without the appropriate generation and handling of cryptographic keys, the integrity, confidentiality, and availability of communications can be severely affected. The following section explains the major threats to the usage of weak keys in security applications.

A. Brute Force Attacks

Brute force attacks try to discover a valid key by trying out different possible key combinations [8]. A successful brute force attack allows an attacker to find the cryptographic key, thereby breaking into the cryptosystem. Brute force attacks can be prevented by using strong keys of sufficient length for secure cryptographic operations. Strong keys of sufficient length hinder the attacker's ability to launch a brute force attack by using a large key space, and making it impossible for the attacker to recover the key in a reasonable amount of time.

B. Rainbow Table Attacks□

Rainbow table attacks try to break the cryptographic key by pre-computing hash values for different key combinations [6-7]. These attacks are launched by constructing rainbow tables that store a hashed value for every word in the key dictionary. A cryptographic key can be broken by looking up the hash values and their corresponding plaintext key in the rainbow table.

III. Threat Model

A brute force attack is a very common methodology adopted by adversaries to break cryptosystems with strong cryptographic operations but weak keys; whereby instead of finding weaknesses in the encryption system, the attacker tries to crack the cryptographic key used for performing the cryptographic operations. The attacker tries each possible key combination to find the correct key that has been employed for carrying out the cryptographic operations [6]. From an attacker's perspective, longer keys are harder to break compared to shorter keys, since the resources required to launch a brute force attack grow exponentially with an increase in key size. The notion of a successful brute force attack is to find the key that has been used to perform ciphering operations, thereby deciphering all the encrypted data using the found key to recover the plaintext [8]. Figure 1 depicts a scenario whereby the attacker's try to break the key used in embedded system devices for secure cryptographic operations.

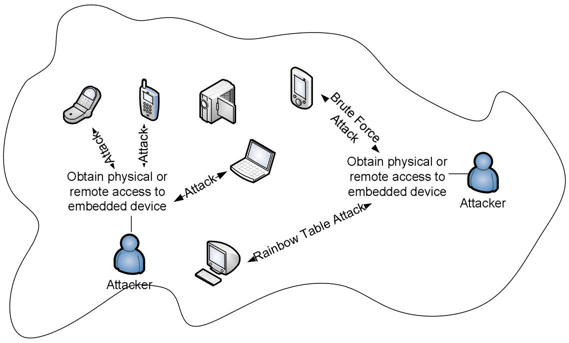
Attack
Attack
Brute Force Attack
Obtain physical or remote access to embedded device
Attack
Obtain physical or remote access to embedded device
Attacker
Rainbow Table Attack
Attacker

Figure 1. Attacks on ICMetrics Keys

Rainbow table attacks pre-compile a series of hashes and their corresponding keys for key lookup. Each rainbow table entry is started with a randomly selected input and its corresponding hash. Rainbow table entries are computed by chains of hash functions and reduction functions that transform a random input value to its final hash value [6]. The purpose of a hash function is to hash a plaintext to generate a hash value, while a reduction function does the reverse to transform a hashed value to its respective plaintext; both of these features enable storage of the pre-computed hashed keys in a compact manner [18]. However a reduction function is not the inverse of a hash function, so computation of a reduction function on a hash value generates a completely new plaintext value. Rainbow tables store the starting plaintext and final hash value after performing multiple iterations of hash and reduction functions. Rainbow table attacks are more efficient as compared to brute force attacks and are therefore able to crack the cryptographic keys quickly. Rainbow tables store pre-computed hashes for all possible key combinations, so that the key can be efficiently cracked by hash and corresponding key lookup.

IV. ICmetrics-Integrated Circuit metrics

ICmetrics or Integrated Circuit metrics makes use of system level characteristics to provide identification to the system [3]. It generates keys based on the hardware/ software characteristics and specification of the node. ICmetrics compute the required metrics on those hardware and software characteristics that are difficult for the attacker to deduce. These metrics/ features are not static but infact vary in a pre-determined fashion. For example, the address and value from the data transactions of a processor; its program address; and metrics for the effectiveness of the program and data caches derived from performance counters, etc [4].

After each key generation stage the produced encryption key is temporary and exists only locally, and the reproduction of the key once again takes place from measurable characteristics of the integrated circuit [2]. ICmetrics is not dependent on any particular encryption algorithm and there are no secret keys to share between the sender and receiver. ICmetrics generates an encryption key directly from measurable properties of a given hardware device, similar to the way biometrics extracts human features to perform an operation.

Analogous to biometrics, the features extracted from integrated circuits might not be stable, in that case the feature values may be based on values taken from a Gaussian distribution [5].

V. Strong ICMetrics Session Key Pair Generation Scheme

The generated ICMetrics key cannot serve as a useful key for cryptographic operations in its raw form, since it might be too short or have low entropy, thereby making it very easy for the attacker to guess and break into the system. Therefore in [11-12], we present an approach for the generation of strong ICMetrics session key pairs, that generates key pairs possessing properties of sufficient key entropy and length. The scheme aims to make the weak ICMetrics key suitable for use in secure cryptographic operations.

The strong ICMetrics key pair generation scheme, generates the session key pairs using the ICMetrics basis number fed into SHA-2 based key derivation function that generates high entropy public/private key pair of sufficient levels of security. The design of the SHA-2 based key derivation function forming the basis of the strong key pair generation scheme rests on the ICMetrics secret key and a random session token from the trusted authority. The random session tokens for each entity are generated by trusted parties corresponding to their particular networks. These session tokens are based on random ID's that are assigned by trusted party higher up in the hierarchy from the requesting node. The effect of a random session token is to create a large set of possible keys associated with a particular raw ICMetrics key. The iterations of the key derivation function also lead to a significant increase in number of rounds performed by the attacker to derive the key [17]. The resultant high entropy session key pairs generated are valid for a single communication session. The session based strong ICMetrics public/private key pair generation mechanism has the following features that address the keying issues in secure embedded system applications:

1. To safeguard against issues related to key compromise, the proposed architecture is based on the use of ICMetrics values for key pair generation.
2. As stated above, each node in the network (entity/trusted party) is assigned a random ID that remains valid during a communication session and a completely new random ID is assigned for subsequent communication sessions. The trusted party makes use of this random session ID to generate a session token that helps safeguard against various sorts of cryptanalytic attacks. This random session token also serves to identify/ authenticate the participating entities.
3. The high entropy session key pairs are formed by combining the ICMetrics generated basis number with the session token generated by the Trusted Party. Both of these secret values are combined through a SHA-2 based key derivation function [12].
4. The main challenge with the ICMetrics generated basis number is the entropy and length of the generated secret value. So the scheme generates a high entropy key of sufficient length using SHA-2 based key derivation function [13] with the session token and ICMetrics basis number as input. It further iterates through multiple rounds of the SHA-2 based hash function to stretch the secret value to the required length, thereby also generating a key with high entropy that is resistant against brute force and rainbow table attacks [18-19].
5. Lastly the key pair generation scheme generates a public key corresponding to the generated high entropy private key by computing the Hermite Normal Form of the private key [1]. The Hermite Normal Form is particularly suitable for public key generation since its unique, non-reversible and doesn't require any random values for operations.

VI. Attack Scenarios

In the following section we use two scenarios to further elaborate on the threat model of a brute force and rainbow table attacks launched on weak ICMetrics keys and further how the scheme for the generation of strong ICMetrics session key pairs safeguards from these threats.

A. *Resilience against Brute Force Attacks*

The first major threat to cryptographic keys is the possibility of being cracked by brute force attacks. The adversary's goal is to break the cryptographic key using software which tries different possible character combinations in quick succession, until the correct key is found. The brute force algorithm uses a trial and error technique, which tries out several key combinations to find the key used for cryptographic operations, so that the adversary is able to decrypt all information destined to the embedded system device. To aid the attacker in the launch a brute force attack, the attacker makes use of a high-performance computer. The attacker machine tries out a large number of key combinations at a very fast pace to recover the key in a short period of time. The high entropy ICMetrics session key pair generation scheme has the ability to resist against brute force attacks, since the generated strong keys have high entropy and sufficient length (256 bits or 512 bits).

The first strength of the scheme that helps safeguard the keys from brute force attacks is the use of random 128 bits device ID assigned to a participating device for the duration of any particular communication session. This random value acts as a salt in the SHA-2 based key derivation and stretching function employed for the generation of high entropy ICMetrics key pairs, as shown in figure 2. The generated high entropy ICMetrics keys are hash values of either 256 bits or 512 bits in length, and are computed by going through multiple iterations of SHA-2 based key stretching and key derivation function. These features of length, entropy and randomness directly translate into more time being taken to break the key, and making brute force attack impossible to launch. Moreover since the computed high entropy ICMetrics key pair only lasts for a single communication session, the attacker has very little chance of breaking the 256/512 bits high entropy keys during that time. Once the session is over, completely new high entropy key pairs are computed and used for cryptographic operations. So the adversary has to start the attack all over again.

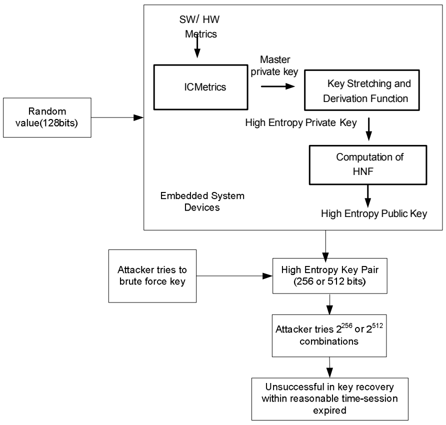

Figure 2. Threat Model-Brute Force Attack

B. *Resilience against Rainbow Table Attacks*

In the second scenario of cracking an ICmetrics strong private key, the attacker tries to launch a rainbow table attack on the hashed strong ICmetrics private key. To launch a rainbow table attack, the attacker generates a large database of all possible key combinations and their corresponding calculated hash values to launch a rainbow table attack. Depending on the length of the key and the employed character set, the attacker pre-computes many hash values and then stores the key and corresponding calculated hash value in the form of rainbow tables sorted by their hash value. If the key employed in the cryptosystem is too long or uses a character not part of the character set known to the attacker, then the generated rainbow tables are completely useless and the key cannot be cracked with those generated rainbow tables. To successfully launch a rainbow table attack on the hashed strong ICMetrics private key, the adversary searches the rainbow table for the hashed key and its corresponding plaintext key. If the hashed key is found in the rainbow table, then the attacker is successful in finding the original key; otherwise the attacker is unsucessful since the hashed value might not be present in the rainbow table.

A major problem with unsalted key hashes in the possibility of a rainbow table attack being launched, that can break the password in an instant, since a hashed key always corresponds to a specific original key. The high entropy ICMetrics session key pair generation scheme has the ability to resist against rainbow table attacks, since the scheme employs a random 128 bit device ID for computation of strong ICMetrics key pairs, which proves to be a very effective method for combating rainbow table attacks. The use of a salt value in the strong ICMetrics session key pair generation scheme makes pre-computed rainbow tables useless, since this means the adversary having to compute a separate rainbow table for each random 128 bit value, which also requires knowledge of the salt value being used during that particular communication session. This process of pre-computation of rainbow tables for salted hashes is much slower; since it first requires the adversary to find salt values specific to the communication session, and then further to compute the rainbow tables with a 128 bit random value at run time.

Table 1. Effect of Random ID on Generated Session Key Pairs

Device	*ICMetrics Basis No. of Device*	*Random Device ID*	*Generated High Entropy session Private Key*	*Generated High Entropy Public Key Pair*
A	**1a29n**	**2638weh23b**	**0x4r238n4nfm**	**0xhrhiwheeru**
B	**defg8**	**U4893bb454**	**0xjerhhewkr9**	**0xertkherk48**
C	**q2np6**	**nkr349061e**	**0x3nrje83jrk**	**0x9347yhtg9u**
A	**1a29n**	**f893rlm087**	**0x5h5nrfhrnf**	**0xhh474urhf5**
A	**1a29n**	**34jghif981**	**0xjgjfur8er8**	**0x8rne38793h**
B	**defg8**	**rjndfbe83n**	**0xwrhriuy34y**	**0x4y4yu4rhri**

As example illustrated in table 1, all the devices forming a part of the network are assigned a random ID by the Trusted Third party that remains valid for the duration of the session. A salt value is associated with an ICMetrics basis key based on the random ID assigned by the trusted third party. Both the ICMetrics basis number and the random partial key assigned by the TTP are combined and fed to the SHA-2 key derivation function, as shown in figure 3. So the randomly generated session ID for every device in the network randomizes the partial key for the device, thereby generating a completely different ICMetrics private key for every session. This also safeguards against rainbow table attacks, since for every session an ICMetrics basis number for any particular device will always correspond to a different hash value as shown in table 1. The public key is then computed by calculating the Hermite Normal Form of the session private key.

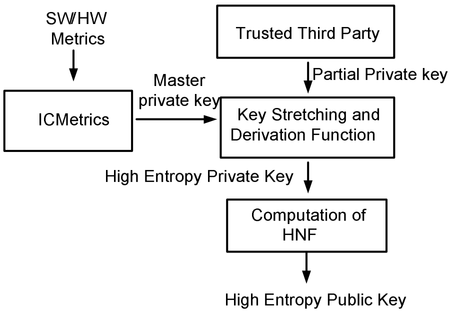

Figure 3. High Entropy Key Pair Generation Scheme [10]

Unless the attacker knows the random ID, he/she cannot extract the private key and therefore the pre-computed rainbow table turn out to be completely useless. For the attacker, trying to break the salted ICMetrics key increases the time and complexity thereby making the strong ICMetrics session key pairs impractical to break, as shown in figure 4.

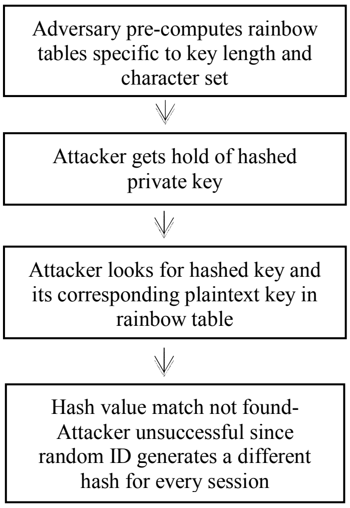
Adversary pre-computes rainbow tables specific to key length and character set
Attacker gets hold of hashed private key
Attacker looks for hashed key and its corresponding plaintext key in rainbow table
Hash value match not found- Attacker unsuccessful since random ID generates a different hash for every session

Figure 4. Threat Model-Rainbow Table Attack

VII. CONCLUSION

ICMetrics is a key technology that generates the cryptographic key for the device based on its hardware and software characteristics. However the generated ICMetrics key suffers from weaknesses of low entropy and insufficient length and is highly susceptible to brute force and rainbow table attacks. In this paper we analyze the security properties of the scheme for the generation of high entropy ICMetrics session key pairs, and further elaborate on how it strengthens the ICMetrics keys while helping safeguard against brute force and rainbow table attacks. The SHA-2 based key stretching and key derivation function that is part of the proposed scheme helps increase the entropy and length of the ICMetrics key, thereby making it safe for use in cryptographic applications. This also safeguards the key from brute force attack since the attacker is not able to break longer keys in sufficient time frame. The random ID assigned to a device for every communication session generates a completely new hashed key in every session and therefore hinders the attacker's ability to pre-compute rainbow tables. Therefore using the scheme for the generation of high entropy ICMetrics session key pairs, resilience against brute force attacks and rainbow table attacks is strengthened and overall survivability of the network is also enhanced.

VIII. FUTURE WORK

Our future plan is to test our scheme through experiments and analysis, thus benchmarking the results against existing key generation schemes that provide security against brute force and rainbow table attacks. We expect that our proposed solution will be an efficient solution providing resilience against brute force and rainbow table attacks.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the support of the UK Engineering and Physical Sciences Research Council under grant EP/K004638/1 and the EU Interreg IV A 2 Mers Seas Zeeën Cross-border Cooperation Programme – SYSIASS project: Autonomous and Intelligent Healthcare System (project's website http://www.sysiass.eu/).

REFERENCES

[1] G. Shmonin, "Hermite normal form: Computation and applications", http://disopt.epfl.ch/webdav/site/disopt/shared/IntPoints2009, Feb. 24, 2009.

[2] E. Papoutsis, W. G. J. Howells, A. B. T. Hopkins, and K. D. McDonald-Maier, "Integrating Multi-Modal Circuit Features within an Efficient Encryption System", Third International Symposium on Information Assurance and Security, IEEE Computer Society Washington, DC, USA, 2007, pp. 83-88.

[3] E. Papoutsis, W. G. J. Howells, A. B. T. Hopkins, and K. D. McDonald-Maier, "Ensuring Secure Healthcare Communications via ICmetric based Encryption on unseen Devices", Symposium on Bio-inspired, Learning and Intelligent Systems for Security, Edinburgh, 20-21 Aug. 2009, pp. 113-117.

[4] E. Papoutsis, W. G. J. Howells, A. B. T. Hopkins, and K. D. McDonald-Maier, "Integrating Feature Values for Key Generation in an ICmetric System," in IEEE NASA/ESA Conference on Adaptive Hardware and Systems (AHS-2009) San Francisco, California, 2009, pp. 82-88.

[5] R. Tahir, K. D. McDonald Maier, "Improving Resilience against Node Capture Attacks in Wireless Sensor Networks using ICMetrics", IEEE Conference on Emerging Security Technologies, Portugal, September 5-7, 2012.

[6] P. Tasevski, "Password Attacks and Generation Strategies", http://home.cyber.ee/ahtbu/CDS2011/PredragTasevskiSlides.pdf

[7] http://kestas.kuliukas.com/RainbowTables/

[8] G. O. Karame, S. Capkun, U. Maurer , "Privacy-Preserving Outsourcing of Brute-Force Key Searches", Proceedings of the 3rd ACM workshop on Cloud Computing Security, New York, USA, 2011, pp.101-112.

[9] F. F. Yao, Y. L. Yin, "Design and Analysis of Password-Based Key Derivation Functions", IEEE Transactions on Information Theory, Vol 51(9), pp. 3292-3297.

[10] R. Tahir, K. D. McDonald Maier, "An ICMetrics based Lightweight Security Architecture using Lattice Signcryption", IEEE Conference on Emerging Security Technologies, Portugal, September 5-7, 2012.

[11] R. Tahir, H. Hu, D. Gu, G. Howells, K. McDonald-Maier, "A Scheme for the Generation of Strong Cryptographic Key Pairs based on ICMetrics", Proceddings of the 7th IEEE Conference on Internet Technology and Secured Transactions, London, UK, December 10-12, 2012.

[12] R. Tahir, H. Hu, D. Gu, G. Howells, K. McDonald-Maier, "A Scheme for the Generation of Strong ICMetrics Session Key Pairs for Secure Embedded System Applications", accepted for publication to the 7th IEEE Symposium on Securirty and Multimodality in Pervasive Environement , Barcelona, Spain, March 26-29, 2013.

[13] B. Kaliski, "PKCS #5: Password-Based Cryptography Specification Version 2.0", RFC 2898, Sept. 2000.

[14] C. Adams, G. Kramer, S. Mister and R. Zuccherato, "On the Security of Key Derivation Functions", Conf. on Industrial Simulation, Spain, pp. 134-145.

[15] J. Kelsey, B. Schneier, C. Hall, and D. Wagner, “Secure Applications of Low-Entropy Keys”, Information Security Workshop (ISW 1997), Japan, pp. 121-134.

[16] NIST's Policy on Hash Functions, National Institute on Standards and Technology Computer Security Resource Center, March 29, 2009.

[17] C. Paar, J. Pelzl, “Hash Functions-Understanding Cryptography, A Textbook for Students and Practitioners", Springer, 2009.

[18] X. Wang, Y. L. Yin, and H. Yu, “Finding Collisions in the Full SHA-1”, 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, pp. 17-36.

[19] B. Schneier, "Schneier on Security: Cryptanalysis of SHA-1", Schneier.com.

[20] “Secure Hash Standard (SHS)”, FIPS PUB 180-3, October 2008, http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf.