Crack me if you can: Hardware Acceleration Bridging the Gap between Practical and Theoretical Cryptanalysis?

A Survey

Mustafa Khairallah
School of Physical and Mathematical Sciences, NTU
Singapore
mustafam001@e.ntu.edu.sg

Zakaria Najm
Temasek Labs at NTU
Singapore, TU Delft
zakaria.najm@ntu.edu.sg

Anupam Chattopadhyay
School of Computer Science and Engineering, NTU
Singapore
anupam@ntu.edu.sg

Thomas Peyrin
School of Physical and Mathematical Sciences, NTU
Singapore
thomas.peyrin@ntu.edu.sg

ABSTRACT

Cryptanalysis is an essential part of cryptology. Not just is it useful to break ciphers for malicious applications, but it is also the basis for building secure ones. In fact almost all the ciphers still in use are trusted to be secure mainly due to the fact that many cryptanalysts are trying hard to break them publicly and failing. However, most of the time successful cryptanalytic results end up violating the cipher designers claims, but the attack itself remains theoretical due to the lack of enough resources/algorithms to efficiently implement it. For example, while the first practical SHA-1 collision was found in 2017, most of the ideas and vulnerabilities behind the attack had been discovered in 2005. The internet and IT industries didn't give much attention to the early theoretical results and it wasn't until 2016 that internet browsers starting getting rid of SHA-1. The leap from 2005 to 2017 was due to advancements in the attack algorithms, implementation techniques and hardware fabrication technologies. While hardware fabrication so far keeps on improving according to Moore's law, the other two aspects require a lot of research effort. In this survey, we touch on several examples of these efforts over the years. The survey is divided into three parts, cryptanalytic attacks designed with specific implementation requirements, previous cryptanalytic machines and quantum computers, the technology that promises to change how we think about cryptography and cryptanalysis.

ACM Reference Format:
Mustafa Khairallah, Zakaria Najm, Anupam Chattopadhyay, and Thomas Peyrin. 2018. Crack me if you can: Hardware Acceleration Bridging the Gap between Practical and Theoretical Cryptanalysis?: A Survey. In *Proceedings of International Conference On Embedded Computer Systems: Architectures, Modeling And Simulation (SAMOS'18).* ACM, New York, NY, USA, 6 pages.

1 INTRODUCTION

While computing machines, hardware acceleration technologies and cryptography have always been closely related, the use of hardware digital circuits to accelerate cracking secure ciphers even predates the invention of the Silicone transistor. The events surrounding World War II led to the emergence of Modern Cryptography, where it changed from an art exclusive to military and intelligence personnel, to a mathematical discipline practiced and studied by mathematicians and computer scientists, as well. Moreover, as it was discovered many years later, it was during the World War II at Bletchley Park in England that the first automated hardware cryptanalytic machines was built. In 1939, Alan Turing designed a mechanical machine called "Bombe", which performed statistical attacks on encrypted German messages, using Enigma [7], intercepted by the British Navy. Later, between 1943 and 1945, British code breakers built another machine called "Colossus", in order to break another German cipher, Lorenz. Unlike Bombe, which was an electro-mechanical machine, Colossus used vacuum tubes and Boolean functions and is considered as the first programmable, electronic and digital computer [39].

In this survey, the modern and potentially future hardware machines for cryptanalytical purposes are discussed. We only consider cases where a single machine is built in order to either break a cipher directly or perform a huge computational task that is considered a milestone towards breaking one. Distributed cryptanalysis projects over many locations are not considered. Besides, we consider only logical attacks that don't include assumptions on the cipher implementations. Hence, side-channel and fault-injection attacks are also not considered, regardless of their costs. In Section 2, a set of cryptanalytic techniques that require specific assumption when it comes to the execution platform are discussed. In Section 3, several examples of implementations or designs of cryptanalytic hardware machines are provided, showing how new and advancing technologies push the limits of what was previously considered impractical. In Section 4, a brief discussion on the effect of quantum computing and quantum attacks on existing ciphers is provided, while the survey is concluded in Section 5.

2 CRYPTANALYTIC ATTACKS WITH TIGHT HARDWARE REQUIREMENTS

Most of the cryptanalytic attacks on ciphers, especially symmetric key ciphers (block ciphers, stream ciphers, hash functions, etc) consist of at least one phase of executing a complex search algorithm. With the huge input and output spaces of the involved functions, this step is very expensive in terms of time and/or memory consumption. Hence, acceleration algorithms and machines usually target efficient and parallelisable implementations of this step. In [25], the authors provided three assumptions that cover a wide variety of attacks and help develop efficient accelerators:

(1) Cryptanalytic algorithms are parallelisable.
(2) Different nodes need to communicate with each other only for a very limited amount of time.
(3) Since the target algorithms are computationally intensive, the communication with the host is very limited compared to the time spent on the computational tasks.

In this section, we describe how some of the costly attacks can be adjusted in order to satisfy these conditions and lead to efficient hardware accelerators. For a wider exploration of different cryptanalytic techniques, we refer the interested reader to any of these resources: [19, 42, 48].

2.1 Brute-Force Attacks

Brute-Force attacks play an important role in the security of ciphers, especially in the field of symmetric key cryptography. Brute-force attacks refer to attacks where all the possible values of a secret variable are tried until the correct value (or a set of valid values) is reached. This type of attack is applicable to any cryptosystem and provides an upper bound on the computational complexity of breaking a cipher. For example, since the Data Encryption Standard (DES) uses a secret key of 56 bits, it requires at most 2^{56} encryptions/decryptions in order to significantly narrow down the space of possible secret keys. Hence, it was believed when DES was introduced, that it has a security level of 56 bits. Any attack that requires less than 2^{56} encryptions/decryptions is considered a genuine threat to the cipher security. Moreover, while 2^{56} operations was considered beyond the realm of possibility in the 1980s and early 1990s, the NIST organization has recently announced that any security level below 112 will be considered insecure from now on [1]. However, implementing brute force attacks is not as straightforward as it sounds and it's practicality is not just subject to the availability of resources. A lot of challenges face the attackers when it comes to memory management, parallelisation and data sorting/searching.

2.2 Time-Memory-Data Trade-off Attacks

In order to overcome the high time complexity required by most attacks, there is a trade-off to be made between the time and space requirements. For example, considering a block cipher $E_K(p)$, which represents a family of bijective permutations parametrized by the key value K, the attacker can choose on one end of the trade-off to ask for $E_K(0)$ and use brute-force to try all possible keys until he finds the correct key (or set of keys, as depending on the size of p and K it may not be possible to find a unique solution using only a single encryption), or the attacker can pre-compute and sort $E_{K'}(0)\ \forall K' \in 2^{|K|}$, then for any instance of the block cipher the key recovery takes $O(1)$ as it involves only one memory access. However, the space complexity of the later case is $O(2^{|K|})$. TMDT attacks try to find a sweet spot between these two extremes, making complex attacks more practical for implementations. In this section we describe two of the famous examples for such attacks.

Meet in the Middle Attacks. The MitM attack was introduced by Diffie and Hellman in 1977 [13]. It applies to scenarios were an intermediate value during a function execution can be represented as the output of two independent random mappings. The most famous example for a cipher vulnerable to this attack is 2DES, which consists of applying the Data Encryption Standard (DES) twice using two independent keys, i.e.,

$$C = E_{K_2}(E_{K_1}(P))$$

A straightforward brute force attack would require 2^{112} operations, since DES has a key size of 56 bits. However, the MitM attack requires only 2^{57} operations and works as follows. First, we notice that

$$I = E_{K_1}(P) = E^{-1}_{K_2}(C)$$

Second, we notice that I is a collision between two random mappings. Hence, finding a pair (K_1, K_2), such that $E_{K_1}(P) = E^{-1}_{K_2}(C)$, falls under the birthday problem and requires only $2^{\frac{|K_1|+|K_2|}{2}}$, i.e., 2^{56} iterations, on average. Since, every iterations consists of one encryption and one decryption operations, the overall number of function calls is 2^{57}, which is only twice the brute force complexity against DES. On the other hand, as most birthday attacks, the MitM requires a huge memory space, since every execution has to be stored and compared to all previous executions until the collision is found.

Hellman Time-Space Trade-off. This attack, first proposed by Hellman in 1980 [18], targets accelerating brute force attacks. For simplicity, we consider only the case where the key size equals the plaintext size. However, the same attack can be generalized to other cases. The attacker chooses a plain-text P, which he knows will be encrypted at some point in the future. Then, he selects a set of possible keys as the starting point of his computation. For example, he can use the set $\{K_i | K_i \in [0, N]\}$, where N is one of the parameters of the time-space trade-off. The next step is to compute $C_i^0 = E_{K_i}(P)$. The attacker iterates over the computation $C_i^j = E_{C_i^{j-1}}(P)$, where $j \in [1, S]$ (S is the second parameter of the trade-off). All the previous step are done offline, without communicating with the target user. The attacker at this point ends up with N chains, each has $S + 1$ nodes, as follows:

$$\begin{aligned} C_0^0 \to C_0^1 \to C_0^2 \cdots \to C_0^S \\ C_1^0 \to C_1^1 \to C_1^2 \cdots \to C_1^S \\ \vdots \\ C_N^0 \to C_N^1 \to C_N^2 \cdots \to C_N^S \end{aligned}$$

The previous lists include $(S + 1) * (N + 1)$ Keys. In order to save memory, the attacker stores only the initial key K_i and the final value C_i^S. Next, in the online phase, the attacker intercepts a ciphertext $C = E_{K^*}(P)$. First, he compares C to the $N + 1$ final values in his pre-computed table. If $C = C_i^S$, since $C = E_{K^*}(P)$, then $E_{K^*}(P) = E_{C_i^{S-1}}(P)$. The attacker returns $K^* = C_i^{S-1}$. Otherwise, the attacker sets $C^0 = C$ and computes the list in the same manner as the lists generated during the offline phase:

$$C^0 \to C^1 \to C^2 \cdots \to C^S$$

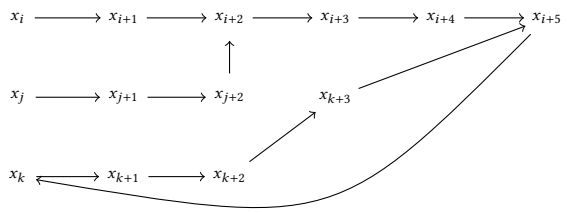

Figure 1: A simplified functional graph example. An edge goes from vertex a to vertex $H^{'}(a)$

If $C^l = C_i^S$, then $E_{K^*}(P) = E_{C_i^{S-l-1}}(P)$. In both cases, the attacker needs to recompute the chain where the collision occurred until the key value is found. On average, the online phase requires $S/2$ operations to find the collision and $S/2$ operations to find the key. However, since the attack covers only $(S+1)*(N+1)$ key candidates, it has a success probability of $\frac{(S+1)*(N+1)}{|\mathcal{KS}|}$, where $|\mathcal{KS}|$ is the size of the full key space.

2.3 Parallel Birthday Search Algorithms

So far, we have described three different generic attacks against ciphers. All these attacks have one feature in common. A cipher is considered as a random mapping $C = E_{K^*}(P) : \{0,1\}^{|P|} \times \{0,1\}^{|K|} \rightarrow \{0,1\}^{|C|}$. In this section, we consider a wider class of functions; collision-resistant compression functions $T = H(x) : \{0,1\}^n \rightarrow \{0,1\}^t$, such that $n \gg t$ and it is computationally hard to find a pair (x_1, x_2) such that $H(x_1) = H(x_2)$. This class of functions has several applications in the construction of secure hash functions and the cryptanalysis of symmetric key ciphers. For examples, the problem of finding such a collision is helpful to the meet in the middle attack described earlier. It is known that the computational complexity of finding a collision for such a function is upper bounded by the birthday bound $2^{t/2}$. However, the efficient design of a collision search algorithm is not a trivial task, specially if the attacker wants to make use of parallelisation over a set of computing machines. This issue is discussed in details in [50]. First, we look at the problem of designing an efficient algorithm for the birthday search problem on a single processing unit. A straightforward approach is to compute $2^{t/2}$ random instances. With high probability, a colliding pair exists in the list formed by these instances. However, such approach requires $O(2^{t/2})$ memory locations and $O(2^t)$ memory accesses/comparisons. In order to overcome these tight requirements, several attacks with different trade-offs have been proposed, almost all of them share the same property; the function in question is reduced to $T = H^{'}(x) : \{0,1\}^t \rightarrow \{0,1\}^t$, which is treated as a pseudo-random function (PRF). One of the useful ways to represent such a function is using a functional graph, which is a directed graph with 2^t vertices and two vertices x and y with an edge from x to y are connected if $y = H(x)$, as shown in Figure 1.

The collision search problem can be treated as a graph search problem, where he attacker is looking for two edges with the same endpoint but with different start-points. Pollard's rho method [37] helps find a collision in the functional graph with a small memory requirement. The underlying idea is to start at any vertex and perform a random walk in the graph until a cycle is found. Unless the attacker is unlucky to have chosen a starting point that is part of the cycle, he ends up with a graph that resembles the Greek letter ρ and the collision is detected.

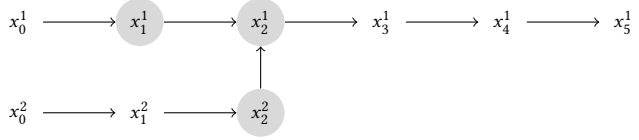

Figure 2: An example of two colliding traces in the functional graph

Nonetheless, Pollard's rho method cannot be efficiently parallelized without modifications. If an attacker tries to run many instances of the algorithms on several machines, independently, each machine will try to look for a cycle in a specific part of the functional graph. However, there is no guarantee that the first colliding pair will be found using a single machine, as each member of the pair can be found using a different machine. For example, two machines can enter the same cycle, but the attacker will not detect this event. Hence, if he uses m machines, he will need $O(\frac{2^t}{\sqrt{m}})$ time to find the collision, as opposed to his original target of $O(\frac{2^t}{m})$.

In [50], the authors proposed a method to achieve $O(\frac{2^t}{m})$ speed-up, using limited memory and communication requirements. First, the attacker defines a distinguished point to be a point $H^{'}(x)$ that has a special property which can be easily checked, e.g. the first d bits are equal to 0. Second, the attacker chooses m random messages $x_0^1, x_0^2, \cdots, x_0^m$ and assigns one of them to each machine. Third, each machine i computes a trace $(x_0^i, x_1^i, \cdots, x_d^i)$, where $x_j^i = H^{'}(x_{j-1}^i)$ and x_d^i is a distinguished point. This is a random walk in the functional graph. If the probability of the condition $x_d^i = H^{'}(x_{d-1}^i)$ is θ, the average length of the trace d is $\frac{1}{\theta}$. Hence, if θ is too large, the traces are too short and the total number of traces $\approx \frac{2^k}{\theta}$ is in the same order of magnitude of 2^k. This means that the attacker does not observe a significant reduction in the memory usage compared to a random-search based algorithm. On the other hand, if θ is too small, the traces become so long, and there is a risk of hitting a cycle within the trace, without ever hitting a distinguished point. Hence the choice of θ is crucial for the attack efficiency. Moreover, it is a good practice to abort the trace after it becomes too long, e.g. $\frac{20}{\theta}$. The previous two steps are repeated $\frac{2^{t/2}\theta}{m}$ times. Each trace is specified in the memory as (x_0^i, d, x_d^i). Finally, a central server needs to sort these traces efficiently, according to the values x_d^i and find the traces that have the same endpoint. Once two similar traces are found, it is easy to find a collision within them that looks like in Figure 2.

3 HARDWARE MACHINES FOR BREAKING CIPHERS

3.1 Brute Force Machines

In this section we describe cases where engineers have been able to build machines to efficiently execute brute force attacks against certain ciphers.

Table 1: Some of the attacks performed using the COPACOBANA platform

Cipher	Attack Type	Time Consumed
DES	Brute Force	6.4 days
A5/1	Guess and Determine [24]	6 hours
ECC (k=79)	Discrete-Log	3.06 hours
ECC (k=97)	Discrete-Log	93.4 days

Deep Crack. In 1998, the Electronic Frontier Foundation (EFF) built a dedicated hardware machine consisting of 1856 ASIC chips connected to a single PC. The machine was able to test over 90 billions DES keys per second, which means that it can go over all the 2^{56} DES keys in 9 days. It was able to solve one of the RSA security DES challenges in 56 hours. The project costed 250,0000 US Dollars and was motivated by the discrepancy between the estimates of academia and government officials regarding the cost and time required to break DES [14].

COPACOBANA. In CHES 2006, COPACOBANA [25] was introduced as an FPGA cluster architecture consisting on 120 FPGAs controlled by a host computer. It is considered to be the first publicly reported configurable platform built specifically for cryptanalysis as its main purpose. The design philosophy behind the architecture depends on the three main assumptions in Section 2. These assumptions are satisfied by both brute force and cryptanalytic attacks. Hence, the COPACABANA has been used to accelerate several attacks [16]. We sum up some of these attacks in Table 1. Some of the attacks performed on the COPACOBANA platform, e.g. guess and determine attack on A5/1, were specially designed in the first place to make use of hardware acceleration [24].

WindsorGreen. In 2016, a document was release accidentally on the New York University server, describing a custom made supercomputer designed by the NSA and IBM, which is believed to have mainly two applications, cracking ciphers and forging cryptographic signatures [2]. However, limited information is available publicly on the project.

3.2 Acceleration of Collision Attacks on Hash Functions

In 2005, the first theoretical collision attack on SHA-1 was published by Wang et al [53]. Since then, a lot of efforts have been targeted towards making the attack more efficient. These efforts are summarized in Figure 3. In 2015, the authors of [17] provided an estimation for finding near collisions on SHA-1, which is a critical step in the collision attacks. The authors provided a design of an Application-Specific Instruction-set Processor (ASIP), named Cracken, which executes specific parts of the attack. It was estimated that to execute the free-start collision and real collision attacks from [43], the attacks will take 46 and 65 days and cost 15 and 121 million Euros respectively.

In 2016, the first attack was practically executed using a cluster of 64 GPUs and took 10 days [45]. Later in 2017, the first real SHA-1 collision was computed [44], using a combination of CPUs and GPUs, taking 6500 CPU years and 100 GPU years. However, the exact details of the machine used were not revealed in the paper.

3.3 The Factoring Machine

The problem of efficiently finding the prime factors of an integer is one of the oldest mathematical problems in the field of Number Theory. It is also the basis of some of the Public Key Cryptosystems (PKC), such as RSA. If a computer can factor $n = pq$ into p and q, it would lead to breaking RSA systems of key size $|n|$. The Number Field Sieve (NFS) algorithm is one of the famous algorithms for solving the factoring problem. However, efficient and cheap implementations of this algorithm are non-trivial. In [49], the author describes three different architectures for machine to perform the NFS algorithm, the cheapest of which costs 400,000 US Dollars and is estimated to take under one year to break RSA-768 in under one year. However, the results are highly speculative and are not supported by any actual implementation.

3.4 Molecular Computers

In [6], Boneh et al. describe an attack on DES that is estimated to take one day to recover the key. It is based on a theorized underlying DNA computer and can be extended to any cipher of key size $\leq$ 64 bits. However, The attack has not been implemented in real life and remains a theoretical idea, until the required DNA computer becomes available.

3.5 Blockchain Mining

While the topic of blockchain mining is not directly related to cryptanalysis or breaking ciphers (specifically hash functions), it is closely related to the acceleration of brute force attacks. The mining operation involves finding an input block to a secure hash function such that the output tag is less than a specific value, i.e. has a certain number of leading 0's. The number of required leading 0's defines the complexity of the problem, which is equivalent to a pre-image attack against a truncated version of the hash function. If that version of the hash function is pre-image resistant, then the mining step is equivalent to a brute force pre-image attack. As the blockchain gets older, the mining step gets more complex. Hence, several industrial players have been interested into accelerating these computations. In 2016, Intel applied for a patent for a Bitcoin mining hardware accelerator [47], which consists of a processor and a coupled hardware accelerator that uses SHA-256 as the main underlying hash function. It is claimed that this system can reduce the power consumption involved in Bitcoin mining by 35%. In April 2018, Samsung has also confirmed that it is building ASIC chips to mine Bitcoin. The new chips utilize the technology and expertise of Samsung's high memory capacity GPUs and can be designed for a specific hash function, to give customers the freedom to choose the target blockchain, not being limited to Bitcoin. However, once fabricated the chip is hash-function specific. It is supposed to increase the power efficiency by 30% and to execute 16 Tera hashes per second [30].

4 QUANTUM COMPUTERS

In recent years, there has been a substantial amount of research on quantum computers. If large-scale quantum computers are ever built, they will be able to break many of the public-key cryptosystems currently in use.

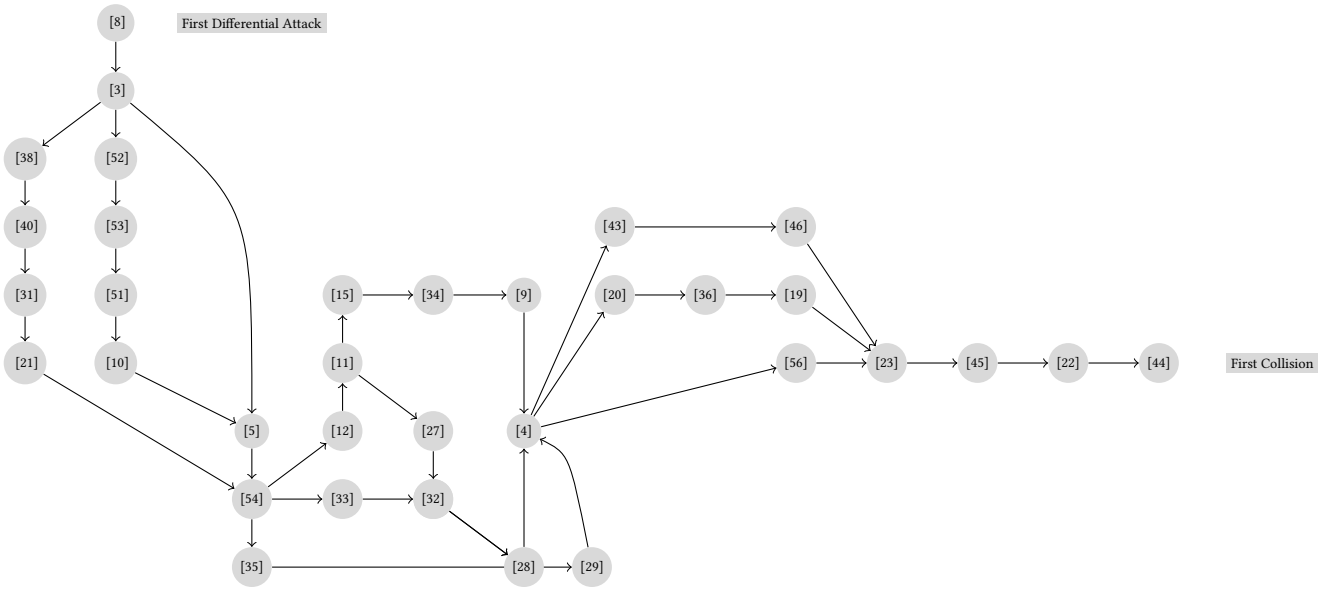

Figure 3: Summary of the improvement and efforts towards accelerating the collision attacks against SHA-1 between 2005 and 2017

The basis of this problem is that the hidden subgroup problem (HSP) is solvable in finite Abelian groups in quantum polynomial time. Thus, factorization, discrete logarithm, discrete logarithm in elliptic curves are solvable in quantum polynomial time as well [41].

The question of when a large-scale quantum computer will be built is a not obvious. While in the past it was less clear that large quantum computers are a physical possibility, many scientists now believe it to be merely a significant engineering challenge. It has taken almost two decade to deploy the currently used public key cryptography standards. That's why, to anticipate, the NIST recently published a call for the post-quantum cryptography standard.

One of the challenge to solve is to find mechanisms for controlling and manipulating the quantum bit easier. Currently a laser is used to physically move each individual qbit stored in the form of ion from one location to another. Breaking cryptograhic standards used today would require to move million of ions at the same time, so millions of lasers, making it impractical for current technologies. But recent advances showed that with a new technic from the university of Sussex called blueprint [26] allows to manipulate thousands of qubits with currently available technoligies. With this technology, a large scale quantum computer consisting of millions of ions would occupy a space the size of a football field, costing upwards of $120 million .

5 CONCLUSION

Breaking a real world cipher in practice is a scientific and technological challenge. When both advances in science and technology gives signs that a cipher can be broken in practice, new cryptographic standards are pulled from the current knowledge. This should be done way before the attack is made practical to absorb the inertia needed to deploy a new cryptographic standard. From a theoretical attack and a practical one, there is still a gap, where new discoveries can be made, that can push further the knowledge that we have to make even better cryptographic standards. When practical attacks are not well anticipated, this inertia can make possible, practical attacks on cryptographic standards that are still in use, which can be catastrophic.

ACKNOWLEDGMENTS

This study is funded by Temasek Laboratories @ NTU.

REFERENCES

[1] Elaine Barker. 2016. Recommendation for key management part 1: General. *NIST special publication* (2016).
[2] Sam Biddle. 2017. NYU ACCIDENTALLY EXPOSED MILITARY CODE-BREAKING COMPUTER PROJECT TO ENTIRE INTERNET. *URL https://theintercept.com/2017/05/11/nyu-accidentally-exposed-military-code-breaking-computer-project-to-entire-internet/* (2017).
[3] Eli Biham and Rafi Chen. 2004. Near-collisions of SHA-0. In *Annual International Cryptology Conference*. Springer, 290–305.
[4] Eli Biham, Rafi Chen, and Antoine Joux. 2015. Cryptanalysis of SHA-0 and Reduced SHA-1. *Journal of Cryptology* 28, 1 (2015), 110–160.
[5] Eli Biham, Rafi Chen, Antoine Joux, Patrick Carribault, Christophe Lemuet, and William Jalby. 2005. Collisions of SHA-0 and Reduced SHA-1. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 36–57.
[6] Dan Boneh, Christopher Dunworth, and Richard J Lipton. 1995. Breaking DES Using a Molecular Computer. *Proceedings of DIMACS workshop on DNA computing* (1995).
[7] Frank Carter. 2008. *The Turing Bombe*. Bletchley Park Trust.
[8] Florent Chabaud and Antoine Joux. 1998. Differential collisions in SHA-0. In *Annual International Cryptology Conference*. Springer, 56–71.
[9] Rafael Chen and Eli Biham. 2011. *New Techniques for Cryptanalysis of Cryptographic Hash Functions*. Ph.D. Dissertation. Computer Science Department, Technion.
[10] Martin Cochran et al. 2007. Notes on the Wang et al. 263 SHA-1 Differential Path. *IACR Cryptology ePrint Archive* 2007 (2007), 474.
[11] Christophe De Canniere, Florian Mendel, and Christian Rechberger. 2007. Collisions for 70-Step SHA-1: on the full cost of collision search. In *International Workshop on Selected Areas in Cryptography*. Springer, 56–73.

[12] Christophe De Canniere and Christian Rechberger. 2006. Finding SHA-1 characteristics: General results and applications. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 1–20.

[13] Whitfield Diffie and Martin E Hellman. 1977. Special feature exhaustive cryptanalysis of the NBS data encryption standard. *Computer* 10, 6 (1977), 74–84.

[14] John Gilmore. 1998. Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design.

[15] Evgeny A Grechnikov. 2010. Collisions for 72-step and 73-step SHA-1: Improvements in the Method of Characteristics. *IACR Cryptology ePrint Archive* 2010 (2010), 413.

[16] Tim Güneysu, Timo Kasper, Martin Novotnỳ, Christof Paar, and Andy Rupp. 2008. Cryptanalysis with COPACOBANA. *IEEE Trans. Comput.* 57, 11 (2008), 1498–1513.

[17] Muhammad Hassan, Ayesha Khalid, Anupam Chattopadhyay, Christian Rechberger, Tim Güneysu, and Christof Paar. 2015. New asic/fpga cost estimates for sha-1 collisions. In *Digital System Design (DSD), 2015 Euromicro Conference on*. IEEE, 669–676.

[18] Martin Hellman. 1980. A cryptanalytic time-memory trade-off. *IEEE transactions on Information Theory* 26, 4 (1980), 401–406.

[19] Antoine Joux. 2009. *Algorithmic cryptanalysis*. CRC Press.

[20] Antoine Joux and Thomas Peyrin. 2007. Hash functions and the (amplified) boomerang attack. In *Annual International Cryptology Conference*. Springer, 244–263.

[21] Charanjit S Jutla and Anindya C Patthak. 2005. A Matching Lower Bound on the Minimum Weight of SHA-1 Expansion Code. *IACR Cryptology ePrint Archive* 2005 (2005), 266.

[22] Pierre Karpman. 2016. *Analyse de primitives symétriques*. Ph.D. Dissertation. Université Paris-Saclay.

[23] Pierre Karpman, Thomas Peyrin, and Marc Stevens. 2015. Practical free-start collision attacks on 76-step SHA-1. In *Annual Cryptology Conference*. Springer, 623–642.

[24] J Keller and B Seitz. 2012. A Hardware-Based Attack on the A5/1 Stream Cipher (2001). *URL http://pv. fernuni-hagen. de/docs/apc2001-final. pdf. Accessed April* (2012).

[25] Sandeep Kumar, Christof Paar, Jan Pelzl, Gerd Pfeiffer, and Manfred Schimmler. 2006. Breaking ciphers with COPACOBANA–a cost-optimized parallel code breaker. In *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 101–118.

[26] Bjoern Lekitsch, Sebastian Weidt, Austin G Fowler, Klaus Mølmer, Simon J Devitt, Christof Wunderlich, and Winfried K Hensinger. 2017. Blueprint for a microwave trapped ion quantum computer. In *Science Advances*, Vol. 3. American Association for the Advancement of Science, e1601540.

[27] Stéphane Manuel. 2008. Classification and generation of disturbance vectors for collision attacks against SHA-1. (2008).

[28] Stéphane Manuel. 2011. Classification and generation of disturbance vectors for collision attacks against SHA-1. *Designs, Codes and Cryptography* 59, 1-3 (2011), 247–263.

[29] Stéphane Manuel and Thomas Peyrin. 2008. Collisions on SHA-0 in one hour. In *International Workshop on Fast Software Encryption*. Springer, 16–35.

[30] Nick Marinoff. 2018. Samsung Is Building ASIC Chips for Halong Mining. *URL https://bitcoinmagazine.com/articles/samsung-building-asic-chips-halong-mining/* (2018).

[31] Krystian Matusiewicz and Josef Pieprzyk. 2006. Finding good differential patterns for attacks on SHA-1. In *Coding and Cryptography*. Springer, 164–177.

[32] Cameron McDonald, Philip Hawkes, and Josef Pieprzyk. 2009. Differential Path for SHA-1 with complexity O (252). *IACR Cryptology ePrint Archive* 2009 (2009), 259.

[33] Florian Mendel, Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. 2006. The impact of carries on the complexity of collision attacks on SHA-1. In *International Workshop on Fast Software Encryption*. Springer, 278–292.

[34] Florian Mendel, Christian Rechberger, and Vincent Rijmen. 2007. Update on SHA-1. *rump session of CRYPTO* 2007 (2007).

[35] Yusuke Naito, Yu Sasaki, Takeshi Shimoyama, Jun Yajima, Noboru Kunihiro, and Kazuo Ohta. 2006. Improved collision search for SHA-0. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 21–36.

[36] Thomas Peyrin. 2008. *Analyse de fonctions de hachage cryptographiques*. Ph.D. Dissertation. PhD thesis, University of Versailles.

[37] John M Pollard. 1978. Monte Carlo methods for index computation (ðÍŚŽðÍŚIJðÍŚŠðÍŚÍ). *Mathematics of computation* 32, 143 (1978), 918–924.

[38] Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. 2005. Exploiting coding theory for collision attacks on SHA-1. In *IMA International Conference on Cryptography and Coding*. Springer, 78–95.

[39] Brian Randell. 1982. Colossus: Godfather of the computer. In *The Origins of Digital Computers*. Springer, 349–354.

[40] Vincent Rijmen and Elisabeth Oswald. 2005. Update on SHA-1. In *CryptographersâĂŹ Track at the RSA Conference*. Springer, 58–71.

[41] Mikolos Santha. 2018. Quantum cryptanalysis: How to break some classical cryptosystems with quantum computers?. In *FWS01 Quantum Cryptography*. Centre for Quantum Technologies, NUS Singapore and CNRS IRIF, Universite Paris Diderot France.

[42] Mark Stamp and Richard M Low. 2007. *Applied cryptanalysis: breaking ciphers in the real world*. John Wiley & Sons.

[43] Marc Stevens. 2013. New collision attacks on SHA-1 based on optimal joint local-collision analysis. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 245–261.

[44] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, and Yarik Markov. 2017. The first collision for full SHA-1. In *Annual International Cryptology Conference*. Springer, 570–596.

[45] Marc Stevens, Pierre Karpman, and Thomas Peyrin. 2016. Freestart collision for full SHA-1. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 459–483.

[46] Marc Martinus Jacobus Stevens et al. 2012. *Attacks on hash functions and applications*. Mathematical Institute, Faculty of Science, Leiden University.

[47] Vikram Suresh, Sudhir Satpathy, and Sanu Mathew. 2018. Bitcoin mining hardware accelerator with optimized message digest and message scheduler datapath. US Patent App. 15/274,200.

[48] Christopher Swenson. 2008. *Modern cryptanalysis: techniques for advanced code breaking*. John Wiley & Sons.

[49] Eran Tromer and ŒćŒÍŒ§ ŒŸŒÍŒŢŒđŒÍ. 2007. *Hardware-based cryptanalysis*.

[50] Paul C Van Oorschot and Michael J Wiener. 1999. Parallel collision search with cryptanalytic applications. *Journal of cryptology* 12, 1 (1999), 1–28.

[51] Xiaoyun Wang. 2006. Cryptanalysis of hash functions and potential dangers. In *Invited Talk at the Cryptographer's Track at RSA Conference 2006*.

[52] Xiaoyun Wang, Andrew C Yao, and Frances Yao. 2005. Cryptanalysis on SHA-1. In *Cryptographic Hash Workshop hosted by NIST*.

[53] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. 2005. Finding collisions in the full SHA-1. In *Annual international cryptology conference*. Springer, 17–36.

[54] Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin. 2005. Efficient collision search attacks on SHA-0. In *Annual International Cryptology Conference*. Springer, 1–16.

[55] Jun Yajima, Terutoshi Iwasaki, Yusuke Naito, Yu Sasaki, Takeshi Shimoyama, Noboru Kunihiro, and Kazuo Ohta. 2008. A strict evaluation method on the number of conditions for the SHA-1 collision search. In *Proceedings of the 2008 ACM symposium on Information, computer and communications security*. ACM, 10–20.

[56] Jun Yajima, Yu Sasaki, Yusuke Naito, Terutoshi Iwasaki, Takeshi Shimoyama, Noboru Kunihiro, and Kazuo Ohta. 2007. A new strategy for finding a differential path of SHA-1. In *Australasian Conference on Information Security and Privacy*. Springer, 45–58.