The First Collision for Full SHA-1

Marc Stevens[1(✉)], Elie Bursztein[2], Pierre Karpman[1], Ange Albertini[2], and Yarik Markov[2]

[1] CWI Amsterdam, Amsterdam, The Netherlands
info@shattered.io
[2] Google Research, Mountain View, USA
https://shattered.io

Abstract. SHA-1 is a widely used 1995 NIST cryptographic hash function standard that was officially deprecated by NIST in 2011 due to fundamental security weaknesses demonstrated in various analyses and theoretical attacks.

Despite its deprecation, SHA-1 remains widely used in 2017 for document and TLS certificate signatures, and also in many software such as the GIT versioning system for integrity and backup purposes.

A key reason behind the reluctance of many industry players to replace SHA-1 with a safer alternative is the fact that finding an actual collision has seemed to be impractical for the past eleven years due to the high complexity and computational cost of the attack.

In this paper, we demonstrate that SHA-1 collision attacks have finally become practical by providing the first known instance of a collision. Furthermore, the prefix of the colliding messages was carefully chosen so that they allow an attacker to forge two distinct PDF documents with the same SHA-1 hash that display different arbitrarily-chosen visual contents.

We were able to find this collision by combining many special cryptanalytic techniques in complex ways and improving upon previous work. In total the computational effort spent is equivalent to $2^{63.1}$ calls to SHA-1's compression function, and took approximately 6 500 CPU years and 100 GPU years. While the computational power spent on this collision is larger than other public cryptanalytic computations, it is still more than 100 000 times faster than a brute force search.

Keywords: Hash function · Cryptanalysis · Collision attack · Collision example · Differential path construction

1 Introduction

A cryptographic hash function $\mathrm{H} : \{0,1\}^* \to \{0,1\}^n$ is a function that computes for any arbitrarily long message M a fixed-length hash value of n bits. It is a versatile cryptographic primitive used in many applications including digital signature schemes, message authentication codes, password hashing and content-addressable storage. The security or even the proper functioning of many of these

applications rely on the assumption that it is practically impossible to *find* collisions, *i.e.* two distinct messages x, y that hash to the same value $\mathrm{H}(x) = \mathrm{H}(y)$. When the hash function behaves in a "sufficiently random" way, the expected number of calls to H (or in practice its underlying fixed-size function) to find a collision using an optimal generic algorithm is $\sqrt{\pi/2} \cdot 2^{n/2}$ (see *e.g.* [33, Appndix A]); an algorithm that is faster at finding collisions for H is then a collision attack for this function.

A major family of hash function is "MD-SHA", which includes MD5, SHA-1 and SHA-2 that all have found widespread use. This family originally started with MD4 [36] in 1990, which was quickly replaced by MD5 [37] in 1992 due to serious attacks [9,11]. Despite early known weaknesses of its underlying compression function [10], MD5 was widely deployed by the software industry for over a decade. The MD5CRK project that attempted to find a collision for MD5 by brute force was halted early in 2004, when Wang and Yu produced explicit collisions [49], found by a groundbreaking attack that pioneered new techniques. In a major development, Stevens *et al.* [45] later showed that a more powerful type of attack (the so-called *chosen-prefix collision attack*) could be performed against MD5. This eventually led to the forgery of a Rogue Certification Authority that in principle completely undermined HTTPS security [46] in 2008. Despite this, even in 2017 there are still issues in deprecating MD5 for signatures [18].

Currently, the industry is facing a similar challenge in the deprecation of SHA-1, a 1995 NIST standard [31]. It is one of the main hash functions of today, and it also has been facing important attacks since 2005. Based on previous successful cryptanalysis [3–5] of SHA-0 [30] (SHA-1's predecessor, that only differs by a single rotation in the message expansion function), Wang *et al.* [48] presented in 2005 the very first collision attack on SHA-1 that is faster than brute-force. This attack, while groundbreaking, was purely theoretical as its expected cost of 2^{69} calls to SHA-1's compression function was practically out-of-reach.

Therefore, as a proof of concept, many teams worked on generating collisions for reduced versions of the function: 64 steps [8] (with a cost of 2^{35} SHA-1 calls), 70 steps [7] (cost 2^{44} SHA-1), 73 steps [15] (cost $2^{50.7}$ SHA-1) and finally 75 steps [16] (cost $2^{57.7}$ SHA-1) using extensive GPU computation power.

In 2013, building on these advances and a novel rigorous framework for analyzing SHA-1, the current best collision attack on full SHA-1 was presented by Stevens [43] with an estimated cost of 2^{61} calls to the SHA-1 compression function. Nevertheless, a publicly known collision still remained out of reach. This was also highlighted by Schneier [38] in 2012, when he estimated the cost of a SHA-1 collision attack to be around US$ 700K in 2015, down to about US$ 173K in 2018 (using calculations by Walker based on a 2^{61} attack cost [43], Amazon EC2 spot prices and Moore's Law), which he deemed to be within the resources of criminals.

More recently, a collision for the full compression function underlying SHA-1 was obtained by Stevens *et al.* [44] using a start-from-the-middle approach and a highly efficient GPU framework (first used to mount a similar

freestart attack on the function reduced to 76 steps [21]). This required only a reasonable amount of GPU computation power, about 10 days using 64 GPUs, equivalent to approximately $2^{57.5}$ calls to SHA-1 on GPU. Based on this attack, the authors projected that a collision attack on SHA-1 may cost between US$ 75K and US$ 120K by renting GPU computing time on Amazon EC2 [39] using spot-instances, which is significantly lower than Schneier's 2012 estimates. These new projections had almost immediate effect when CABForum Ballot 152 to extend issuance of SHA-1 based HTTPS certificates was withdrawn [13], and SHA-1 was deprecated for digital signatures in the IETF's TLS protocol specification version 1.3.

Unfortunately CABForum restrictions on the use of SHA-1 only apply to actively enrolled Certification Authority certificates and not on any other certificates, *e.g.* retracted CA certificates that are still supported by older systems (and CA certificates have indeed been retracted for continued use of SHA-1 certificates to serve to these older systems unchecked by CABForum regulations[1]), and certificates for other TLS applications including up to 10% of credit card payment systems [29,47]. It thus remains in widespread use across the software industry for, *e.g.*, digital signatures of software, documents, and many other applications, most notably in the GIT versioning system.

It is well worth noting that academic researchers have not been the only ones to compute (and exploit) hash function collisions. Nation-state actors [24, 25,34] have been linked to the highly advanced espionage malware "Flame" that was found targeting the Middle-East in May 2012. As it turned out, it used a forged signature to infect Windows machines via a man-in-the-middle attack on *Windows Update*. Using a new technique of *counter-cryptanalysis* that is able to expose cryptanalytic collision attacks given only one message from a colliding message pair, it was proven that the forged signature was made possible by a *then secret* chosen-prefix attack on MD5 [12,42].

2 Our Contributions

We are the first to exhibit an example collision for SHA-1, presented in Table 1, thereby proving that theoretical attacks on SHA-1 have now become practical. Our work builds upon the best known theoretical collision attack [43] with estimated cost of 2^{61} SHA-1 calls. This is an *identical-prefix collision attack*, where a given prefix P is extended with two distinct *near-collision block pairs* such that they collide for any suffix S:

$$\text{SHA-1}\left(P||M_1^{(1)}||M_2^{(1)}||S\right) = \text{SHA-1}\left(P||M_1^{(2)}||M_2^{(2)}||S\right). \tag{1}$$

The computational effort spent on our attack is estimated to be equivalent to $2^{63.1}$ SHA-1 calls (see Sect. 6). There is certainly a gap between the theoretical attack as presented in [43] and our executed practical attack that was based

[1] For instance, SHA-1 certificates are still being sold by CloudFlare at the time of writing: https://www.cloudflare.com/ssl/dedicated-certificates/.

Table 1. Colliding message blocks for SHA-1.

CV_0	4e	a9	62	69	7c	87	6e	26	74	d1	07	f0	fe	c6	79	84	14	f5	bf	45
$M_1^{(1)}$			7f	46	dc	93	a6	b6	7e	01	3b	02	9a	aa	1d	b2	56	0b		
			45	ca	67	d6	88	c7	f8	4b	8c	4c	79	1f	e0	2b	3d	f6		
			14	f8	6d	b1	69	09	01	c5	6b	45	c1	53	0a	fe	df	b7		
			60	38	e9	72	72	2f	e7	ad	72	8f	0e	49	04	e0	46	c2		
$CV_1^{(1)}$	8d	64	d6	17	ff	ed	53	52	eb	c8	59	15	5e	c7	eb	34	f3	8a	5a	7b
$M_2^{(1)}$			30	57	0f	e9	d4	13	98	ab	e1	2e	f5	bc	94	2b	e3	35		
			42	a4	80	2d	98	b5	d7	0f	2a	33	2e	c3	7f	ac	35	14		
			e7	4d	dc	0f	2c	c1	a8	74	cd	0c	78	30	5a	21	56	64		
			61	30	97	89	60	6b	d0	bf	3f	98	cd	a8	04	46	29	a1		
CV_2	1e	ac	b2	5e	d5	97	0d	10	f1	73	69	63	57	71	bc	3a	17	b4	8a	c5

CV_0	4e	a9	62	69	7c	87	6e	26	74	d1	07	f0	fe	c6	79	84	14	f5	bf	45
$M_1^{(2)}$			73	46	dc	91	66	b6	7e	11	8f	02	9a	b6	21	b2	56	0f		
			f9	ca	67	cc	a8	c7	f8	5b	a8	4c	79	03	0c	2b	3d	e2		
			18	f8	6d	b3	a9	09	01	d5	df	45	c1	4f	26	fe	df	b3		
			dc	38	e9	6a	c2	2f	e7	bd	72	8f	0e	45	bc	e0	46	d2		
$CV_1^{(2)}$	8d	64	c8	21	ff	ed	52	e2	eb	c8	59	15	5e	c7	eb	36	73	8a	5a	7b
$M_2^{(2)}$			3c	57	0f	eb	14	13	98	bb	55	2e	f5	a0	a8	2b	e3	31		
			fe	a4	80	37	b8	b5	d7	1f	0e	33	2e	df	93	ac	35	00		
			eb	4d	dc	0d	ec	c1	a8	64	79	0c	78	2c	76	21	56	60		
			dd	30	97	91	d0	6b	d0	af	3f	98	cd	a4	bc	46	29	b1		
CV_2	1e	ac	b2	5e	d5	97	0d	10	f1	73	69	63	57	71	bc	3a	17	b4	8a	c5

on it. Indeed, the theoretical attack's estimated complexity does not include the inherent relative loss in efficiency when using GPUs, nor the inefficiency we encountered in actually launching a large scale computation distributed over several data centers. Moreover, the construction of the second part of the attack was significantly more complicated than could be expected from the literature.

To find the first near-collision block pair $(M_1^{(1)}, M_1^{(2)})$ we employed the open-source code from [43], which was modified to work with our prefix P given in Table 2, and for large scale distribution over several data centers. To find the second near-collision block pair $(M_2^{(1)}, M_2^{(2)})$ that leads to the collision was more challenging, as the attack cost is known to be significantly higher, but also because of additional obstacles.

In Sect. 5 we will discuss in particular the process of building the second near-collision attack. Essentially we followed the same steps as was done for the first near-collision attack [43], combining many existing cryptanalytic techniques. Yet we further employed the SHA-1 collision search GPU framework from Karpman *et al.* [21] to achieve a significantly more cost efficient attack.

We also describe two new additional techniques used in the construction of the second near-collision attack. The first allowed us to use additional differential

Table 2. Identical prefix of our collision.

25 50 44 46 2d 31 2e 33 0a 25 e2 e3 cf d3 0a 0a	%PDF-1.3.%......
0a 31 20 30 20 6f 62 6a 0a 3c 3c 2f 57 69 64 74	.1 0 obj.<</Widt
68 20 32 20 30 20 52 2f 48 65 69 67 68 74 20 33	h 2 0 R/Height 3
20 30 20 52 2f 54 79 70 65 20 34 20 30 20 52 2f	0 R/Type 4 0 R/
53 75 62 74 79 70 65 20 35 20 30 20 52 2f 46 69	Subtype 5 0 R/Fi
6c 74 65 72 20 36 20 30 20 52 2f 43 6f 6c 6f 72	lter 6 0 R/Color
53 70 61 63 65 20 37 20 30 20 52 2f 4c 65 6e 67	Space 7 0 R/Leng
74 68 20 38 20 30 20 52 2f 42 69 74 73 50 65 72	th 8 0 R/BitsPer
43 6f 6d 70 6f 6e 65 6e 74 20 38 3e 3e 0a 73 74	Component 8>>.st
72 65 61 6d 0a ff d8 ff fe 00 24 53 48 41 2d 31	ream......$SHA-1
20 69 73 20 64 65 61 64 21 21 21 21 21 85 2f ec	is dead!!!!!./.
09 23 39 75 9c 39 b1 a1 c6 3c 4c 97 e1 ff fe 01	.#9u.9...<L.....

paths around step 23 for increased success probability and more degrees of freedom without compromising the use of an early-stop technique. The second was necessary to overcome a serious problem of an unsolvable strongly over-defined system of equations over the first few steps of SHA-1's compression function that threatened the feasibility of finishing this project.

As can be deduced from Eq. 1, our example colliding files only differ in two successive random-looking message blocks generated by our attack. We exploit these limited differences to craft two colliding PDF documents containing arbitrary distinct images. Examples can be downloaded from https://shattered.io. PDFs with the same MD5 hash have previously been constructed by Gebhardt *et al.* [14] by exploiting so-called Indexed Color Tables and Color Transformation functions. However, this method is not effective for many common PDF viewers that lack support for these functionalities. Our PDFs rely on distinct parsings of JPEG images, similar to Gebhardt *et al.*'s TIFF technique [14] and Albertini *et al.*'s JPEG technique [1]. Yet we improved upon these basic techniques using very low-level "wizard" JPEG features such that these work in all common PDF viewers, and even allow very large JPEGs that can be used to craft multi-page PDFs. This overall approach and the technical details will be described in a separate article [2].

The remainder of this paper is organized as follows. We first give a brief description of SHA-1 in Sect. 3. Then, we give a high-level overview of our attack in Sect. 4, followed by Sect. 5 that details the entire process and the cryptanalytic techniques employed, where we also highlight improvements with respect to previous work. Finally, we discuss the large-scale distributed computations required to find the two near-collision block pairs in Sect. 6. The parameters used to find the second colliding block are given in the appendix, in Sect. A.

3 The SHA-1 Hash Function

We provide a brief description of SHA-1 as defined by NIST [31]. SHA-1 takes an arbitrary-length message and computes a 160-bit hash. It divides the (padded)

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.