CWI

Centrum Wiskunde & Informatica

Real World Cryptanalysis

Marc Stevens

Cryptology Group

CWI Amsterdam

Real World Cryptanalysis

Real World Cryptanalysis

Cryptographic Standards crucial for secure Internet

Gain confidence in security over time through extensive scrutiny
(Before & After Standardization)

Real World Cryptanalysis

Cryptographic Standards crucial for secure Internet

Occasionally leap in cryptanalysis exposes unknown weaknesses

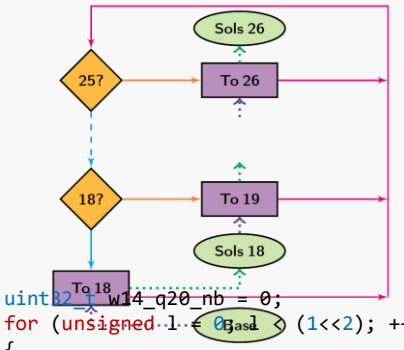
Deprecate Standard !

Real World Cryptanalysis

Theory
Analyze

t	Bits Q_t: $b_{31}\dots b_0$	#
−3	10001010 11111100 01010110 11011110	32
−2	11000100 10011010 01100010 −0−10110	32
−1	01111101 01010011 01101110 −0−11110	32
0	11101011 00011111 000010−+ +0−11010	32
1	..0..0.. ....11.. 0−0−..01 −1−.....	13
2	.1.!0+.. ....1+.. −0++..00 −−+.....	15
3	.1!.01.. .0..+−.. 1−0−.... −−+.....	14
4	!−..−1.. .0..+−!. +1++.... .+.....	13
5	!−00−.00 ^−001−.0 101+0000 1+000000	30
6	!+11−011 ++11−−01 1.+−1111 1.111111	30
7	!1..−... 00.^−−.! −.01.... .1^.^...	15
8	!1..+... 10!−.... −.0−.0.. ..+0+..0	15
9	..!.1... ...010.. −..+.0.. .!001^.0	14
10	00.!−010 00.1..10 .00+!+.0 .01+1−1−	25
11	110.−111 1100^011 01110+01 001−000+	31
12	.11^00+1 0010+1+^ 00^1111. 1−0−0+−0	30
13	^1+−−−0 1−0+0+0− +++++++1 +−−+−++0	32
14	−−1110−+ +++++0+1 00000010 +−−0−−−.	31
15	1+1+1−1− 011−1+10 0000000− 011−.10.	30
16	01...00+ 10111+1. ..+..1.. 100−^01.	21
17	.0.^.+.1 .1.^.+.. ..−..1.^ .0.0..0.	13
18	.......1 .+...+.. ..1..+.. .1.1..1.	8
19	0....^.+ ....0−.. ..−..... .−....−.	8
20	1...0... .^..1−.. 0....^.. .1.....0	9
21	+...1..^ ...0−0.. 1.^..... .0....^0	11
22	....+... ...1.^.. +....... .1.....+	6
23	^......0 ..0−^... 1....... .+.0....	8
24	....^..1 ..10.... 0.....0. ...1...^	8
25	.......− ..−+.... ....^.−....	5
26	..0..... ...−.... .....0+.	4
27	..1....^ ..^1.... .....1+. ...^....	7
28	..+..... ...0.... .....+−.	4
29	..0..... ....0.	2
30	..−..... ....^1.	3
31	..−.....	1
32	....	0
33	..!.....	1
34 − 60	....	0
61	....	
62	.... +.......	
63	.... +.......	
64	.... +.......	

Practice
Build




```
uint32_t w14_q20_nb = 0;
for (unsigned l = 0; l < (1<<2); ++l)
{
    NEXT_NB(w14_q20_nb, W14NBQ20M);
    m14 &= ~W14NBQ20M;
    m14 |= w14_q20_nb;
    q15 += w14_q20_nb;
    q16 += rotate_left(w14_q20_nb, 5);
    m17 ^= rotate_left(m14, 1);
    m20 ^= rotate_left(m17, 1);
    uint32_t w15_q20_nb = 0;
    for (unsigned k = 0; k < (1<<5); ++k)
    {
        NEXT_NB(w15_q20_nb, W15NBQ20M);
        m15 &= ~W15NBQ20M;
        m15 |= w15_q20_nb;
        m18 ^= rotate_left(m15, 1);
        q16 += w15_q20_nb;
```

Demonstrate
Apply

```
%PDF-1.3.%.....
.1 0 obj.<</Widt
h 2 0 R/Height 3
 0 R/Type 4 0 R/
Subtype 5 0 R/Fi
lter 6 0 R/Color
Space 7 0 R/Leng
th 8 0 R/BitsPer
Component 8>>.st
eam.....$SHA-1
is dead!!!!!./.
#9u.9...<L.....
F....~.;.....V.
.g....K.Ly..+=.
.m.i...kE.S....
8.rr/..r..I..F.
0W..........+.5
```

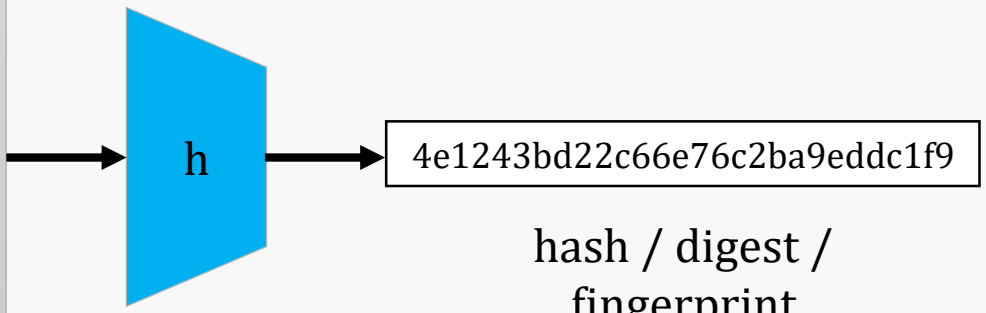
Cryptographic hash functions

Cryptographic hash functions

A hash function is a deterministic mapping

from arbitrary length inputs to a fixed length output

This is an input to a cryptographic hash function. The input is a very long string, that is reduced by the hash function to a string of fixed length. There are additional security conditions: it should be very hard to find an input hashing to a given value (a preimage) or to find two colliding inputs (a collision).



hash / digest / fingerprint

MDC-2
MD2, MD4, MD5
SHA-1
old & weak

RIPEMD-160
SHA-2-{256,512}
secure

SHA-3-{256,512}
New Std in 2015
NIST: use SHA-2

Collision resistance

Collision resistance

Find $m \neq k$ such that $H(m) = H(k)$

Only max. (n/2)-bit security!

128-bit hash ⇒ 64-bit security
160-bit hash ⇒ 80-bit security
256-bit hash ⇒ 128-bit security
512-bit hash ⇒ 256-bit security

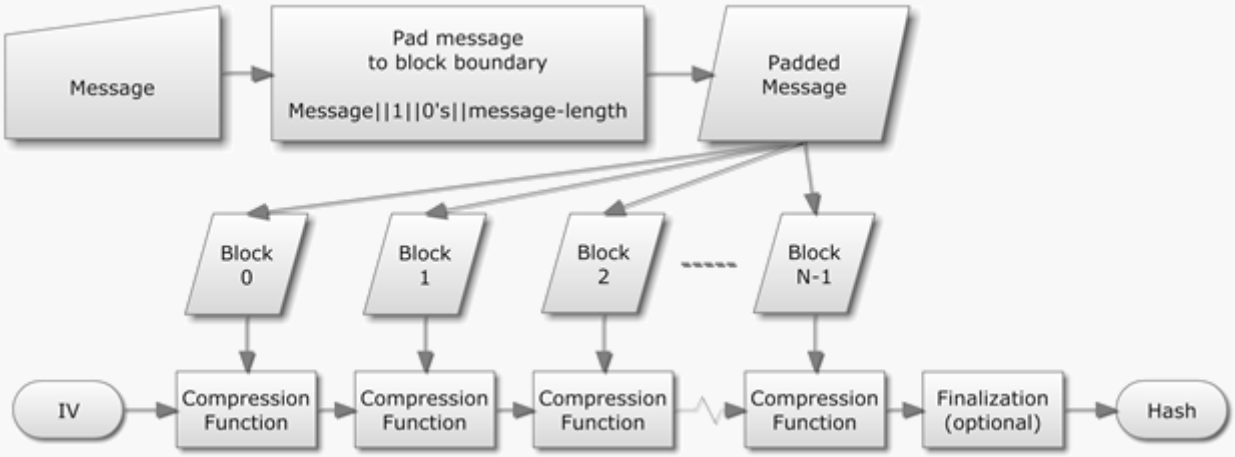
Note:
Bitcoin network computes

- 2^{64} SHA-2 / sec
- $2^{80.5}$ SHA-2 / day
- 2^{84} SHA-2 / 12days
- 2^{89} SHA-2 / year

⇒
breaks 80-bit security
brute-force in 1 day!

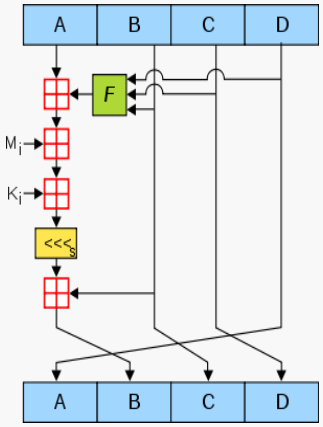
Design of hash functions

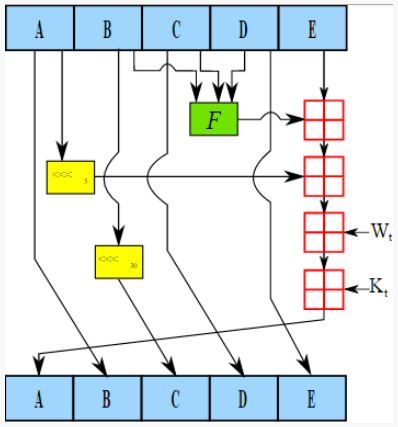
- **Merkle-Damgård Construction**
  - Splits message into 512-bit blocks
  - Processes blocks iteratively using compression function

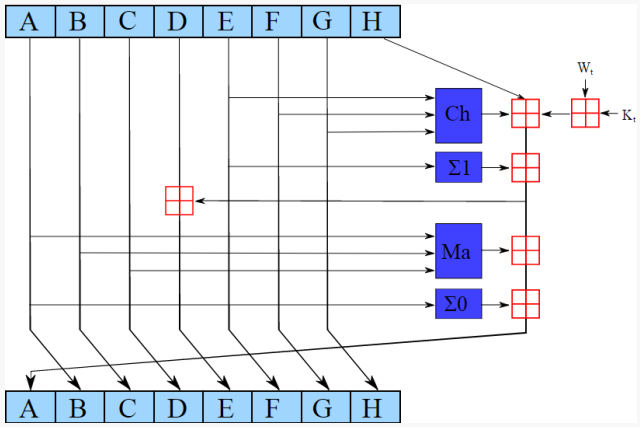


- Security reduction
  - collision hash function $\Rightarrow$ collision compression function

MD5 / SHA-1 / SHA-256 compression function







MD5

very weak

permutation

SHA-1

theoretically weak

linear recurrence

SHA-2

secure

non-linear

message expansion

16 x 32-bit → {64,80} x 32-bit

Differential cryptanalysis

Differential cryptanalysis

- Consider two different instances

  Compress(CV,M)

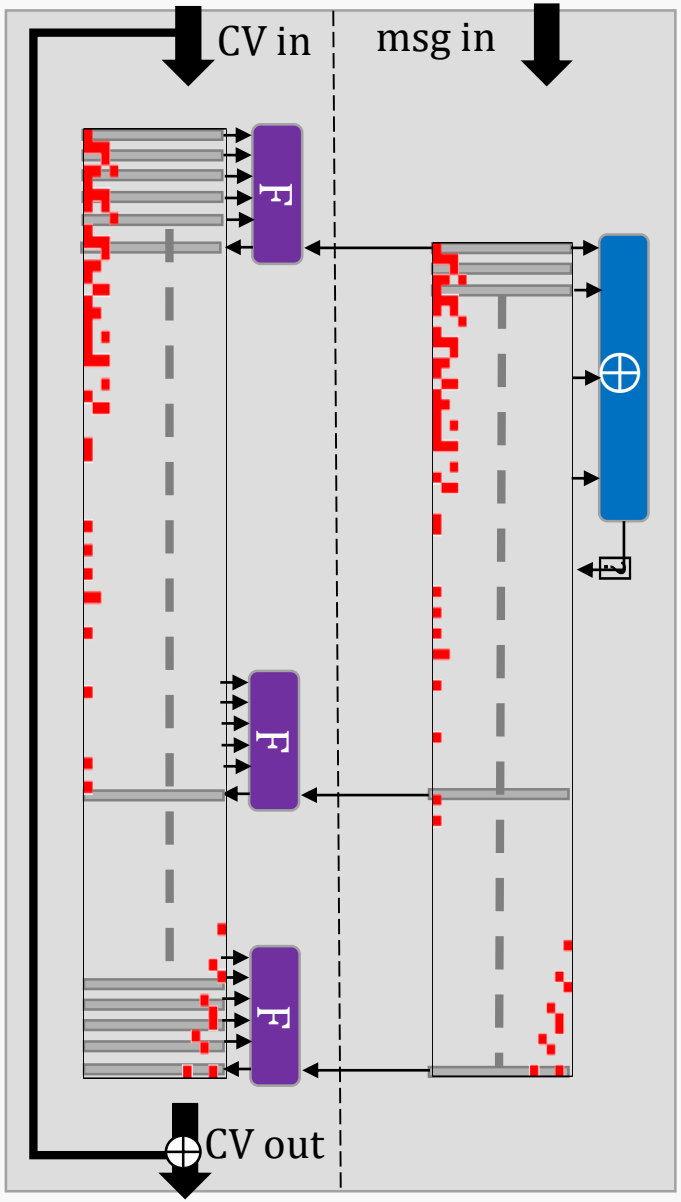
  Compress(CV',M')

- Analyze differences

Differential path

- Precise description of all differences propagating through compression function

- Translate differential path into **system of equations** to solve to find M, M'



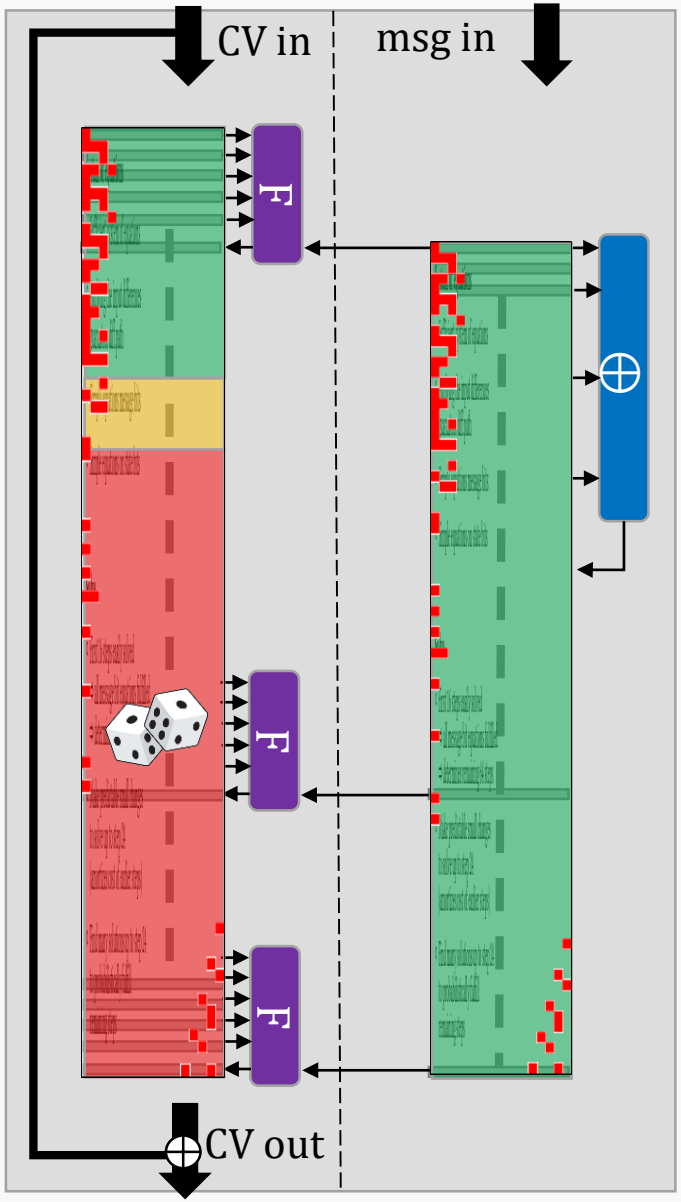
Differential cryptanalysis

System of equations

- Sufficient system of equations:
  Applying the input differences
  guarantees diff.path
- Simple equations
  on message and state bits

Solve

- First 16 steps easily solved
  ⇒ all message bit equations fulfilled
- Make predictable small changes
  to solve up to step 24
  (amortizes cost of earlier steps)
- Probabilistically fulfill
  remaining steps
  (with many solutions up to step 24)



Deprecating MD5 in 2008

(known to be practically broken since 2004)

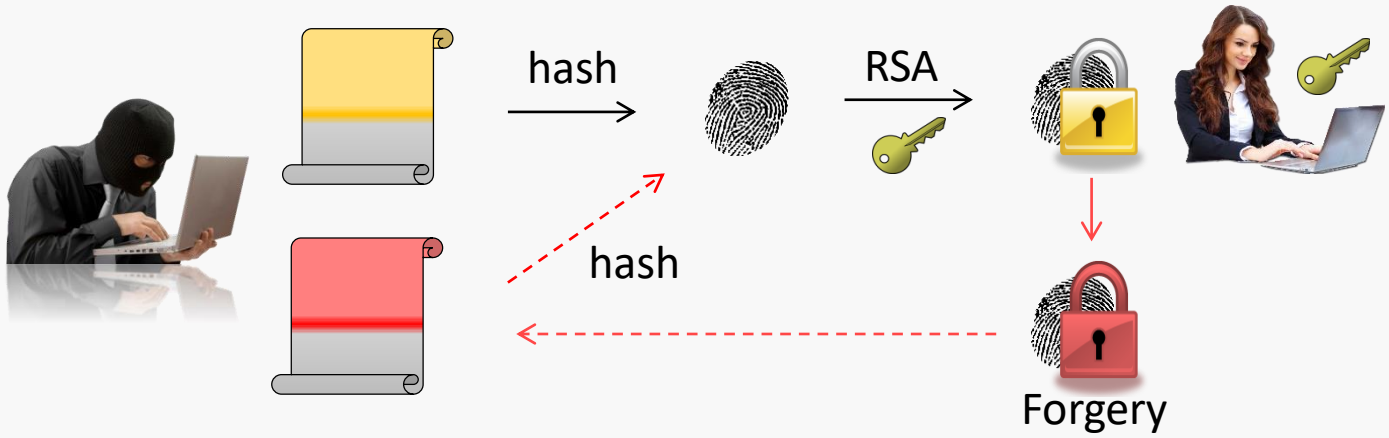
Joint work with:

Alexander Sotirov, Jacob Applebaum, Arjen Lenstra, David Molnar, Dag Arne Osvik, Benne de Weger

Rogue C.A.

2008 [SSALMOdW]: Breakthrough on MD5

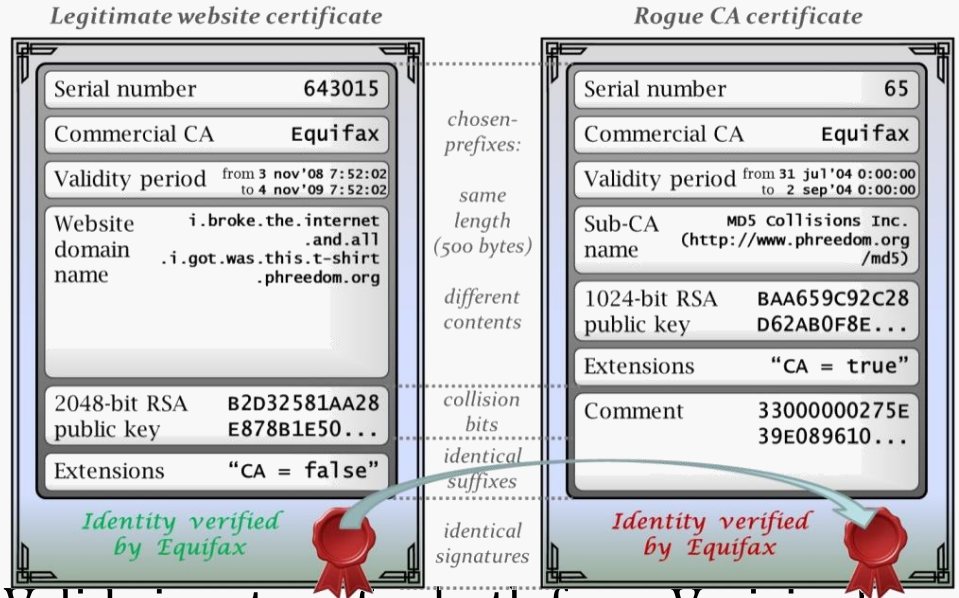
- Practical *chosen-prefix collision* attack on MD5
- Arbitrary different prefixes made to collide



Rogue C.A.

Example chosen-prefix collision between

- Our website: https://i.broke.the.internet.and.all.i.got.was.this.t-shirt.phreedom.org
- A hand-crafted sub-C.A. certificate 😈



- Valid signature for both from Verisign!

Using 200 PlayStation 3s

Rogue C.A.

- Realistic Man-in-the-Middle attack against any secure website

- Responsible disclosure
  - Pre-informed Browsers and C.A.s
  - Rogue C.A. purposely crippled: only valid in August 2004
- MD5 deprecated within hours
- Software released for research
  - Anyone can create chosen-prefix collisions
  - ≈1 day on quadcore machine
  - https://github.com/cr-marcstevens/hashclash

What happened since?

- 2009: CABforum: MD5 deprecated for signatures
- 2012: supermalware Flame uses forged MD5 signature
  to push fake Windows Updates
  Discovery of yet-unknown variant MD5 collision attack
- 2016: SLOTH: Transcript collision attacks against TLS, IKE, SSH
- 2017: Oracle JRE rejects MD5 signatures
  Originally planned for Januari, was postponed till April
- 2018: US SWGDE (Scientific Working Group on Digital Evidence)
  Publication "*explains that the use of the MD5 and SHA1 hash algorithms remains acceptable*"

Deprecating SHA-1 in 2017

(known to be weak since 2005)

Joint work with

Ange Albertini, Elie Bursztein, Pierre Karpman, Yarik Markov

Weaknesses

SHA-1 is not collision resistant

Collision attack with complexity 2^{69} (4M core-years) [WangYY 2005]

Later improved to 2^{61} (15,300 core-years) [Stevens 2012]

Projected costs of SHA-1 collisions [Schneier 2012]

$2.77M in 2012
$700K by 2015
$173K by 2018 ⇒ "we can postpone 5 years.."
$43K by 2021

(based on [Stevens 2012], Amazon EC2 rates & Moore's Law)

Practical SHA-1 collision remained open problem

GPU ≫ CPU

- [S13]: SHA-1 collision attack with complexity $\approx 2^{61}$
- ⇒ CPU attack: 15.3K coreyears

- [SPK16]: attack complexity $\approx 2^{62.2}$ on GTX-970
- ⇒ GPU attack: 112 GPUyears
- ≈ $100k renting fee (on Amazon EC2)
- ×**7** lower cost in 2015 than predicted earlier by Schneier
- Initiated collaboration with Google

First SHA-1 Collision

- **Collaboration with Google [SBKAM17]**
  - Google Infrastructure:
    Large heterogenous cluster of CPUs & GPUs
  - But: no direct access, proprietary Compile & Job system
  - 'Blind' adaptation source-code by Google

  - **First near-collision attack**
    - Took 3583 core years ≈ $\mathbf{2^{60}}$ SHA-1 compressions
    - Run on 100k+ PCs in several weeks

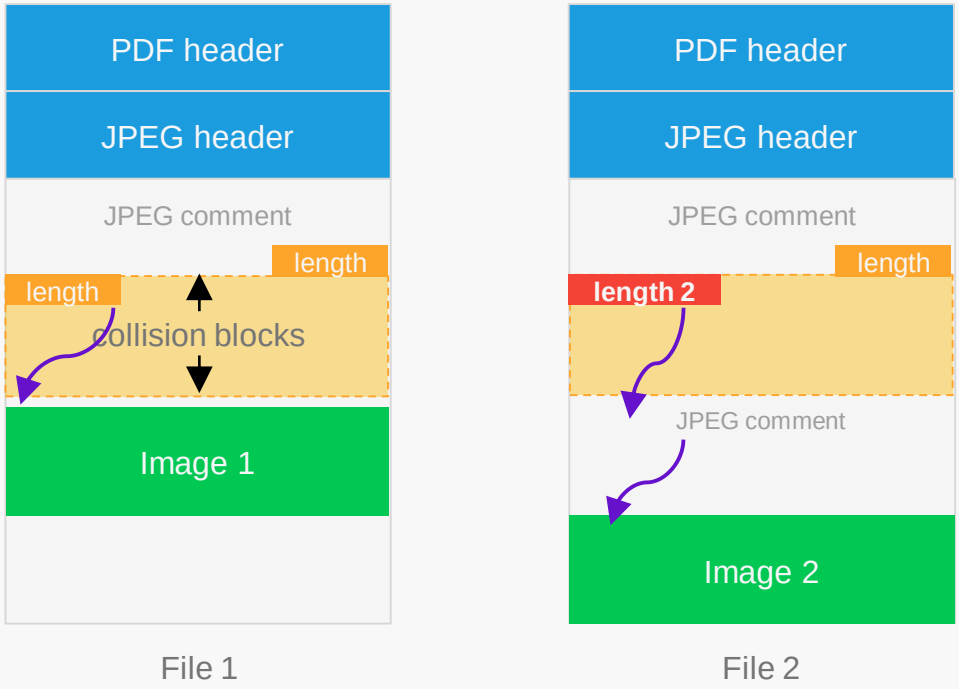
  - **Second near-collision attack**
    - Tailored to 1st NC output
    - Using NVIDIA Tesla K20, K40, K80
    - Took ≈ 114 K20years ≈ 71 K80years ≈ $\mathbf{2^{62.8}}$ SHA-1
    - Run on >3000 GPUs in just 8 calendar days

  - Collision on https://shattered.io/

Colliding PDFs

Reusable meaningful SHA-1 collision:

- 1 collision: infinite colliding PDF-pairs with distinct embedded JPGs
- Use JPG for page-content ⇒ arbitrary distinct page contents
- Use PDF image cropping ⇒ arbitrary distinct multi-page contents



- DIY: https://github.com/nneonneo/sha1collider

Impact

- Project Webpage, Google Drive & Gmail check for SHA-1 collisions

- Unexpectedly collision can break Subversion repositories
  - Webkit developer submitted test to prove WebKit resistant to SHA-1 collisions
  - Broke Webkit repository
  - Internal deduplication uses SHA-1 and keeps only 1 colliding file
  - MD5 is used to check integrity ⇒ will always fail on checkout
- Git started moving away from SHA-1
- Git & GitHub now using strengthened SHA-1 implementation by default

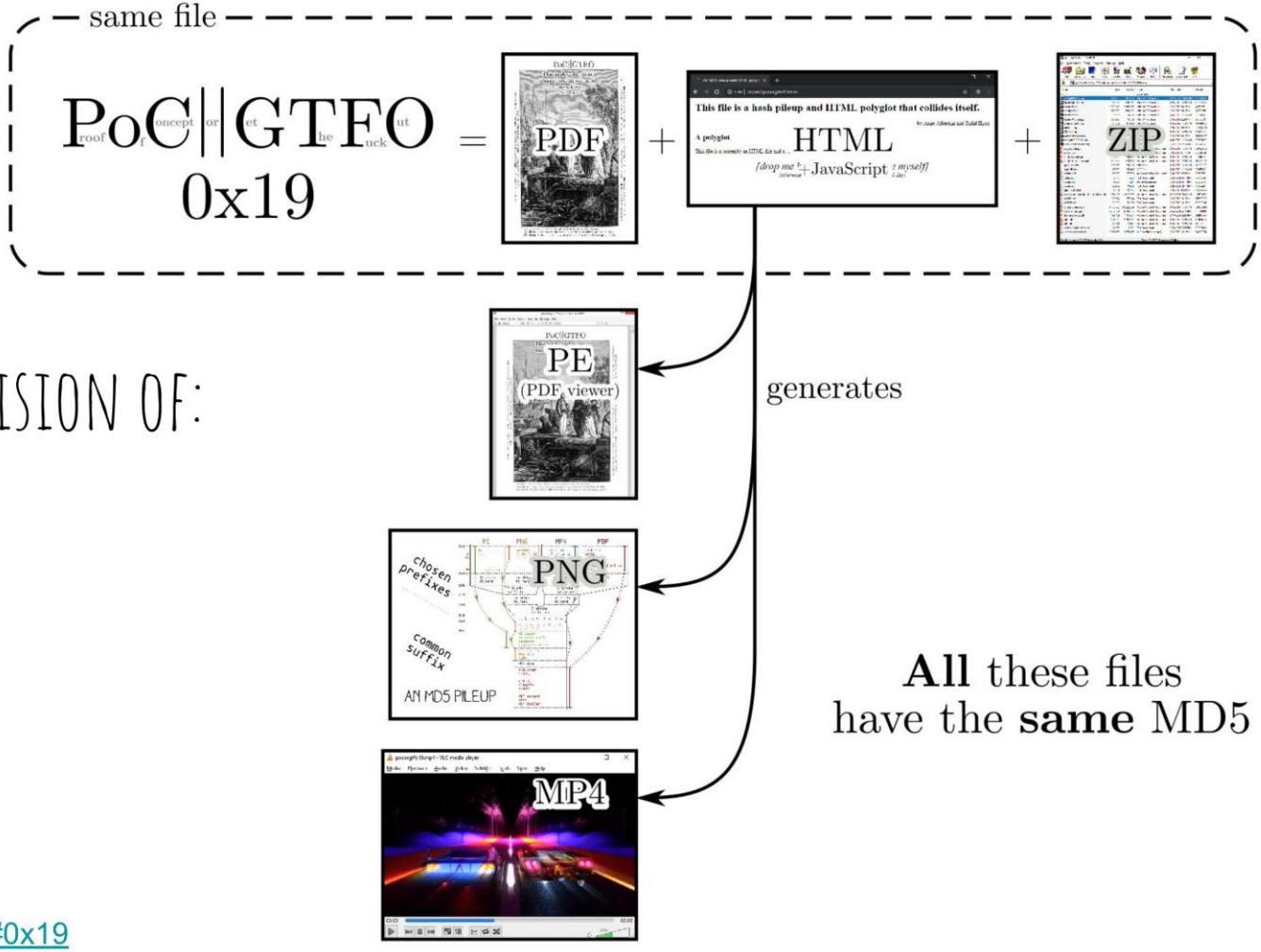
- CA/Browser Forum: Ballot 152
  - Extend issuance SHA-1 certificates up to 1 Jan. 2017 (before: 1 Jan. 2016)
  - (unaltered: deprecate SHA-1 certificates after 1 Jan. 2017)
  - Our recommendations on 8 Oct. ensured Ballot did not pass on 16 Oct.
- TLS 1.3 draft 9
  - Deprecated all uses of SHA-1 digital signatures

From attacks to toys

Instant Collisions

- Instant collision scripts for many file formats
  - Instant, re-usable and generic collisions
  - Take any pair of files, run script, get colliding files
- SHA-1: PDF, HTML
- MD5
  - PDF
  - PNG, JPG, JP2
  - MP4, GIF
  - PE (windows executable)

Toys



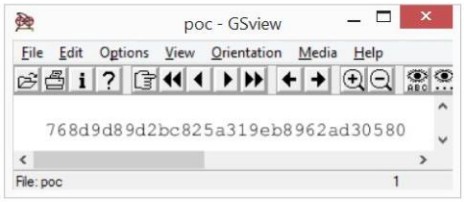
An instant collision of:

- a document
- an executable
- an image
- a video.

https://github.com/angea/pocorgtfo#0x19

Toys

Hashquines: documents that show their own MD5 hash

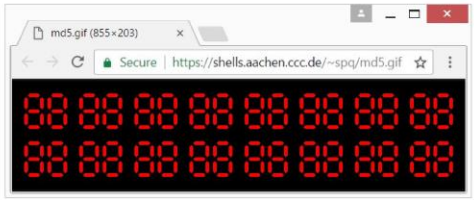


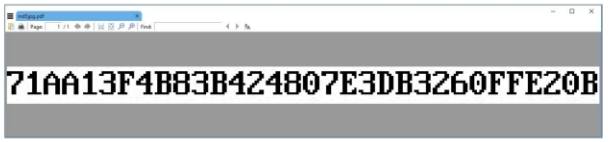
THE FIRST EVER!

PostScript by Greg

ANIMATED

GIFs by spq

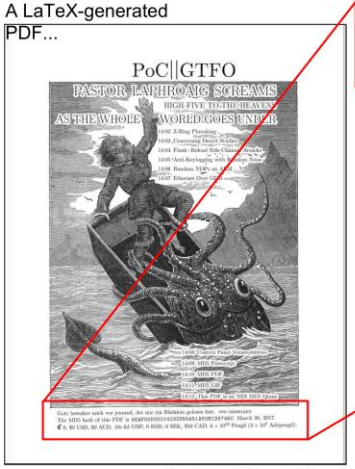




As images

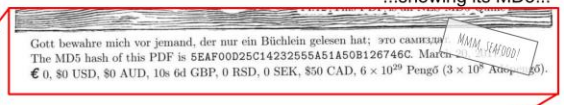
PDFs by Mako

A LaTeX-generated PDF...



(15x32=480 collisions)

...showing its MD5...



...also a NES rom...

...showing the same MD5!
(4x32=128 collisions)

608?

As text

```
$ pdftotext -q md5text.pdf -
66DA5E07C0FD4C921679A65931FF8393
$ md5sum md5text.pdf
66da5e07c0fd4c921679a65931ff8393 md5text.pdf
```

Thank you!