# *SHA-1 is a Shambles*

## *First Chosen-Prefix Collision on SHA-1 and Application to the PGP Web of Trust*

Gaëtan Leurent Thomas Peyrin

Inria, France

NTU, Singapore

Real World Crypto 2020

https://sha-mbles.github.io

# SHA-1

- Hash function designed by NSA in 1995
- Standardized by NIST, ISO, IETF, ...
- Widely used untill 2015

## Cryptanalysis of SHA-1

| | | |
|---|---|---|
| *2005-02* | Theoretical collision with $2^{69}$ op. | [Wang & al., Crypto'05] |
| ... | Several unpublished collision attacks in the range $2^{51} - 2^{63}$ | |
| *2010-11* | Theoretical collision with $2^{61}$ op. | [Stevens, EC'13] |
| *2015-10* | Practical freestart collision (on GPU) | [Stevens, Karpman & Peyrin, Crypto'15] |
| *2017-02* | Practical collision with $2^{64.7}$ op. (GPU) | [Stevens & al., Crypto'17] |

- Levchin prize awarded yesterday to Wang and Stevens for breaking SHA-1 in practice

# SHA-1 Usage in the Real World

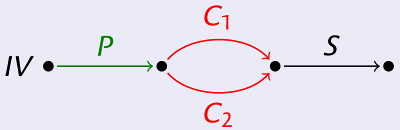
- SHA-1 certificates (X.509) still exists
  - CAs sell legacy SHA-1 certificates for legacy clients
  - Accepted by many non-web modern clients
  - ICSI Certificate Notary: 1.3% SHA-1 certificates
- PGP signatures with SHA-1 are still trusted
  - Default hash for key certification in GnuPGv1 (legacy branch)
  - 1% of public certifications (Web-of-Trust) in 2019 use SHA-1
- SHA-1 still allowed for in-protocol signatures in TLS, SSH
  - Used by 3% of Alexa top 1M servers
- HMAC-SHA-1 ciphersuites (TLS) are still used by 8% of Alexa top 1M servers
- Probably a lot of more obscure protocols...
  - EMV credit cards use weird SHA-1 signatures
  - ...

# Chosen-Prefix Collisions [Stevens, Lenstra & de Weger, EC'07]

- Collisions are hard to exploit: garbage collision blocks $C_i$

## Identical-prefix collision

- Given IV, find $M_1 \neq M_2$ s. t. $H(M_1) = H(M_2)$




- Arbitrary common prefix/suffix, random collision blocks
- Breaks integrity verification
- Colliding PDFs (breaks signature?)

## Chosen-prefix collision

- Given $P_1$, $P_2$, find $M_1 \neq M_2$ s. t. $H(P_1 \| M_1) = H(P_2 \| M_2)$



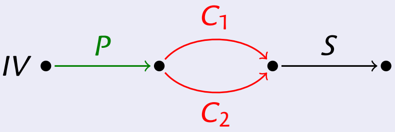

- Breaks certificates
  Rogue CA [Stevens & al, Crypto'09]
- Breaks TLS, SSH
  SLOTH [Bhargavan & L, NDSS'16]

# *Chosen-Prefix Collisions* [Stevens, Lenstra & de Weger, EC'07]

- Collisions are hard to exploit: garbage collision blocks $C_i$

## *Identical-prefix collision*

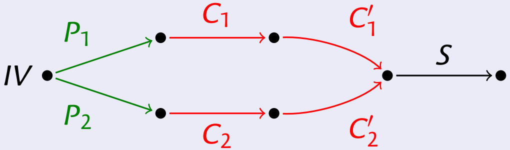
- Given IV, find $M_1 \neq M_2$ s. t. $H(M_1) = H(M_2)$




- Arbitrary common prefix/suffix, random collision blocks
- Breaks integrity verification
- Colliding PDFs (breaks signature?)

## *Chosen-prefix collision*

- Given $P_1, P_2$, find $M_1 \neq M_2$ s. t. $H(P_1 \parallel M_1) = H(P_2 \parallel M_2)$




- Breaks certificates
  Rogue CA [Stevens & al, Crypto'09]
- Breaks TLS, SSH
  SLOTH [Bhargavan & L, NDSS'16]

# Our results

## *Chosen-prefix collision attack on SHA-1*

- Theoretical attack at Eurocrypt 2019
- Practical attack today

1. Complexity improvements (factor $8 \sim 10$)

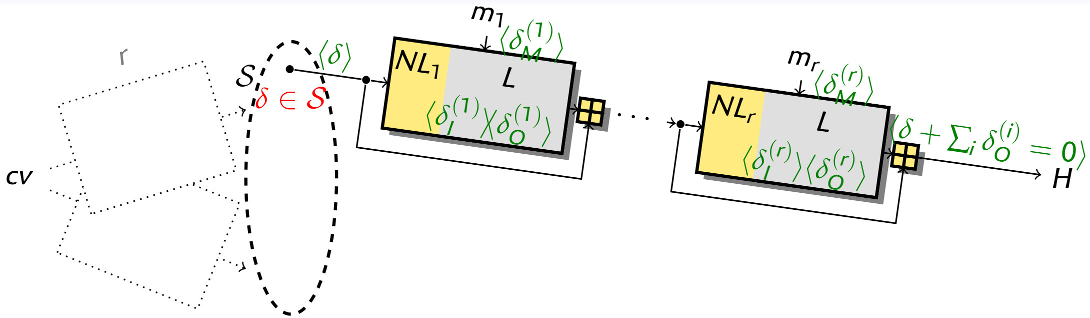
   *identical-prefix collision* from $2^{64.7}$ to $2^{61.2}$ (11 kUS$ in GPU rental)

   *chosen-prefix collision* from $2^{67.1}$ to $2^{63.4}$ (45 kUS$ in GPU rental)

2. Record computation
   - Implementation of the full CPC attack
   - 2 months using 900 GPU (GTX 1060)

3. PGP Web-of-Trust impersonation
   - 2 keys with different IDs and colliding certificates
   - Certification signature can be copied to the second key

# Chosen-prefix collision attack on SHA-1 [L. & P., EC'19]




1. **Setup:** Find a set of "nice" chaining value differences $\mathcal{S}$
2. **Birthday phase:** Find $m_1, m_1'$ such that $H(P_1 \| m_1) - H(P_2 \| m_1') \in \mathcal{S}$
3. **Near-collision phase:** Erase the state difference, using near-collision blocks

- Expected complexity $\approx 2^{67}$ [EC'19]
- After improvements $2^{63} \sim 2^{64}$

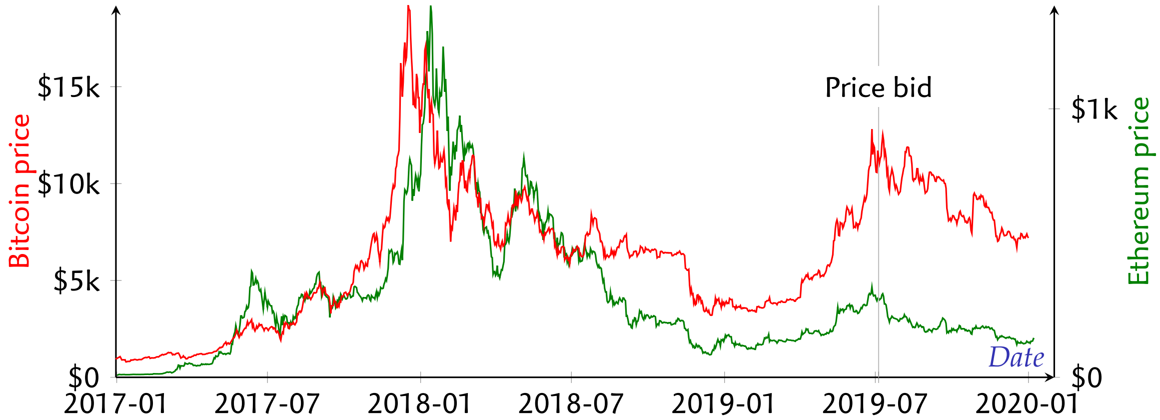
# Running a $2^{64}$ computation on a budget

- Running the attack on Amazon/Google cloud GPU is estimated to cost 160 kUS$ (spot/preemptible instances)
- After cryptocurrency crash in 2018, cheap GPU farms to rent!
  - 👍 3–4 times cheaper
    45 kUS$ with current public prices on gpuserversrental.com
  - 👎 Gaming or mining-grade GTX cards (rather than Tesla)
  - 👎 Low-end CPUs
  - 👎 Slow internet link
  - 👎 No cluster management
  - 👎 Pay by month, not on-demand

- Pricing fluctuates together with cryptocurrencies prices
- We didn't get optimal prices...

# Running a $2^{64}$ computation on a budget

## Bitcoin price history




- Pricing fluctuates together with cryptocurrencies prices
- We didn't get optimal prices...

# Birthday phase

Find $m_1, m_1'$ such that $H(P_1 \parallel m_1) - H(P_2 \parallel m_1') \in \mathcal{S}$

- Set $\mathcal{S}$ of $2^{38}$ "nice" chaining value differences
- Birthday paradox: complexity about $\sqrt{2^n / |\mathcal{S}|} = 2^{61}$
- Chains of iterations to reduce the memory [van Oorschot & Wiener, CCS'94]
  - Truncate `SHA-1` to 96 bits, partial collision likely to be in $\mathcal{S}$
  - About 500GB of storage
  - Easy to parallelize on GPU
  - Expected complexity $\approx 2^{62}$, ($2^{26.4}$ truncated collisions)
- Success after one month
  - $2^{62.9}$ computations ($2^{27.7}$ truncated collisions)
  - Bad luck! ☹

# Near-collision phase

**Erase the state difference, using near-collision blocks**

- Very technical part of the attack: each block similar to a collision attack
  - Find the useful output differences for the next block by exploring $\mathcal{S}$
  - Build a differential trail with specific input/output conditions
  - Build GPU code dedicated to the trail: neutral bits, boomerangs, ...
- For simplicity, we use variants of the core trail of Stevens for all blocks
  - Reuse most neutral bits / boomerang analysis
  - Reuse most GPU code [Stevens, Bursztein, Karpman, Albertini & Markov, C'17]
- Aim for 10 blocks, expected complexity: $2^{62.8}$
  - Last block: $2^{61.6}$ (equivalent to collision attack)
  - Intermediate blocks: $2^{62.1}$ in total (each block is cheap)
- Success after one month
  - $2^{62}$ computations (time lost when preparing the trails and GPU code)
  - Good luck! ☺

# September 27: The First SHA-1 Chosen-prefix Collision

- 416-bit prefix
- 96 birthday bits
- 9 near-collision blocks

| Message A | Message B |
|---|---|
| 99040d047fe81780012000ff4b65792069732070617274206f66206120636f6c<br>6c6973696f6e212049742773206120747261702179c61af0afcc054515d9274e | 99030d047fe81780011800ff50726163746963616c205348412d312063686f73<br>656e2d707265666978206366c6c6973696f6e211d276c6ba661e1040e1f7d76 |
| 7307624b1dc7fb23988bb8de8b575dba7b9eab31c1674b6d974378a827732ff5<br>851c76a2e60772b5a47ce1eac40bb993c12d8c70e24a4f8d5fcdedc1b32c9cf1 | 7f076249ddc7fb332c8bb8c2b7575dbec79eab2be1674b7db34378b4cb732fe1<br>891c76a0260772a5107ce1f6e80bb9977d2d8c68524a4f9d5fcdedcd0b2c9ce1 |
| 9e31af2429759d42e4dfdb31719f587623ee552939b6dcdc459fca53553b70f8<br>7ede30a247ea3af6c759a2f20b320d760db64ff479084fd3ccb3cdd48362d96a | 9231af26e9759d5250dfdb2d4d9f58729fee553319b6dccc619fca4fb93b70ec<br>72de30a087ea3ae67359a2ee27320d72b1b64fecc9084fc3ccb3cdd83b62d97a |
| 9c430617caff6c36c637e53fde28417f626fec54ed7943a46e5f5730f2bb38fb<br>1df6e0090010d00e24ad78bf92641993608e8d158a789f34c46fe1e6027f35a4 | 904306150aff6c267237e523e228417bde6fec4ecd7943b44a5f572c1ebb38ef<br>11f6e00bc010d01e90ad78a3be641997dc8e8d0d3a789f24c46fe1eaba7f35b4 |
| cbfb827076c50eca0e8b7cca69bb2c2b790259f9bf9570dd8d4437a3115faff7<br>c3cac09ad25266055c27104755178eaeff825a2caa2acfb5de64ce7641dc59a5 | c7fb8272b6c50edaba8b7cd655bb2c2fc50259e39f9570cda94437bffd5fafe3<br>cfcac0981252661 5e827105b79178eaa43825a341a2acfa5de64ce7af9dc59b5 |
| 41a9fc9c756756e2e23dc713c8c24c9790aa6b0e38a7f55f14452a1ca2850ddd<br>9562fd9a18ad42496aa97008f74672f68ef461eb88b09933d626b4f918749cc0 | 4da9fc9eb56756f2563dc70ff4c24c932caa6b1418a7f54f30452a004e850dc9<br>9962fd98d8ad4259dea97014db4672f232f461f338b09923d626b4f5a0749cd0 |
| 27fddd6c425fc4216835d0134d15285bab2cb784a4f7cbb4fb514d4bf0f6237c<br>f00a9e9f132b9a066e6fd17f6c42987478586ff651af96747fb426b9872b9a88 | 2bfddd6e825fc431dc35d00f7115285f172cb79e84f7cba4df514d571cf62368<br>fc0a9e9dd32b9a16da6fd16340429870c4586feee1af96647fb426b53f2b9a98 |
| e4063f59bb334cc00650f83a80c42751b71974d300fc2819a2e8f1e32c1b51cb<br>18e6bfc4db9baef675d4aaf5b1574a047f8f6dd2ec153a93412293974d928f88 | e8063f5b7b334cd0b250f826bcc427550b1974c920fc280986e8f1ffc01b51df<br>14e6bfc61b9baee6c1d4aae99d574a00c38f6dca5c153a834122939bf5928f98 |
| ced9363cfef97ce2e742bf34c96b8ef3875676fea5cca8e5f7dea0bab2413d4d<br>e00ee71ee01f162bdb6d1eafd925e6aebaae6a354ef17cf205a404fbdb12fc45 | c2d9363e3ef97cf25342bf28f56b8ef73b5676e485cca8f5d3dea0a65e413d59<br>ec0ee71c201f163b6f6d1eb3f525e6aa06ae6a2dfef17ce205a404f76312fc55 |
| 4d41fdd95cf2459664a2ad032d1da60a73264075d7f1e0d6c1403ae7a0d861df<br>3fe5707188dd5e07d1589b9f8b6630553f8fc352b3e0c27da80bddba4c64020d | 4141fddb9cf24586d0a2ad1f111da60ecf26406ff7f1e0c6e5403afb4cd861cb<br>33e5707348dd5e1765589b83a7663051838fc34a03e0c26da80bddb6f464021d |

# Attacking key certification [Stevens, Lenstra & de Weger, EC'07]


Alice

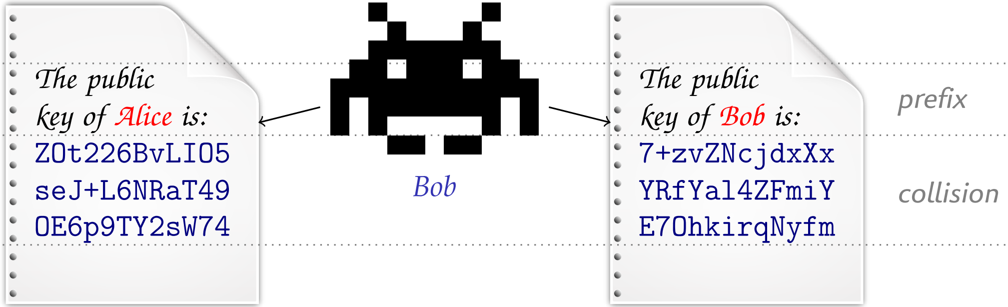
## PKI Infrastructure

- Alice generates key
- Asks CA to sign
- Certificate proves ID

## Impersonation attack

1. Bob creates keys s.t. $H(\texttt{Alice}||k_A) = H(\texttt{Bob}||k_B)$
2. Bob asks CA to certify his key $k_B$
3. Bob copies the signature to $k_A$, impersonates Alice

# Attacking key certification [Stevens, Lenstra & de Weger, EC'07]



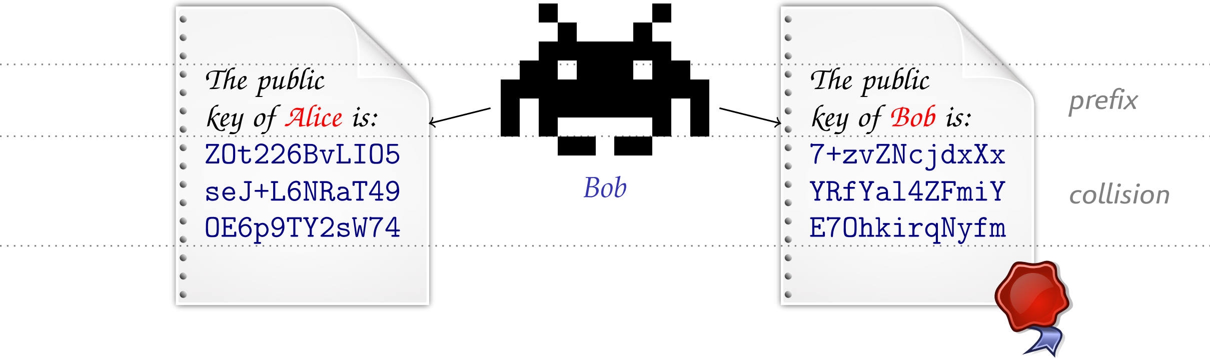

## PKI Infrastructure

- Alice generates key
- Asks CA to sign
- Certificate proves ID

## Impersonation attack

1. Bob creates keys s.t. $H(\texttt{Alice}||k_A) = H(\texttt{Bob}||k_B)$
2. Bob asks CA to certify his key $k_B$
3. Bob copies the signature to $k_A$, impersonates Alice

# Attacking key certification [Stevens, Lenstra & de Weger, EC'07]



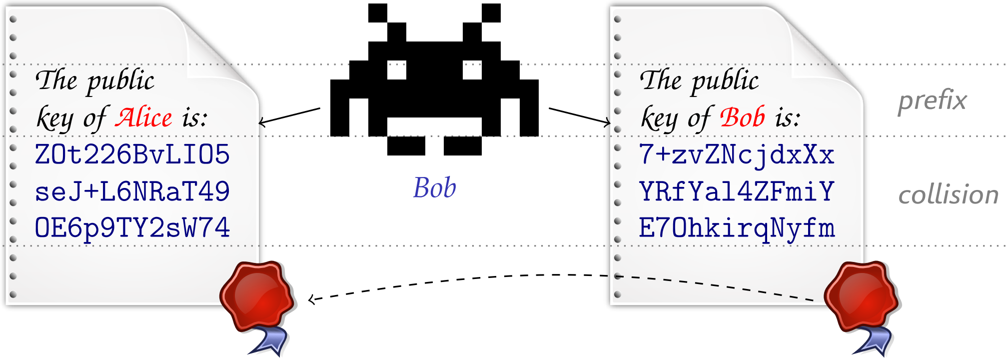

## PKI Infrastructure

- Alice generates key
- Asks CA to sign
- Certificate proves ID

## Impersonation attack

1. Bob creates keys s.t. $H(\texttt{Alice}||k_A) = H(\texttt{Bob}||k_B)$
2. Bob asks CA to certify his key $k_B$
3. Bob copies the signature to $k_A$, impersonates Alice

# Attacking key certification [Stevens, Lenstra & de Weger, EC'07]




## PKI Infrastructure

- Alice generates key
- Asks CA to sign
- Certificate proves ID

## Impersonation attack

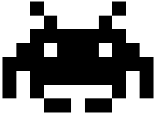
1. Bob creates keys s.t. $H(\texttt{Alice}||k_A) = H(\texttt{Bob}||k_B)$
2. Bob asks CA to certify his key $k_B$
3. Bob copies the signature to $k_A$, impersonates Alice

# PGP identity certificates

- PGP identity certificate has public key first, UserID next
  - Each blob prefixed by length
  - Cannot just use the ID a prefix as with X.509 certificates
  - Quite rigid format (weird extensions not signed)

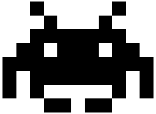
- Use keys of different length, fields misaligned
- PGP format supports for JPEG picture in key, and picture can be signed
  - JPEG readers ignore garbage after End of Image marker

- Certificate A has RSA-8192 public key, with victim ID
- Certificate B has RSA-6144 public key, and attacker's picture
  - Stuff JPEG in key A, and ID B in JPEG
  - Need very small JPEG: example 181-byte JPEG (*almost compliant*)



# PGP identity certificates

- PGP identity certificate has public key first, UserID next
  - Each blob prefixed by length
  - Cannot just use the ID a prefix as with X.509 certificates
  - Quite rigid format (weird extensions not signed)
- Use keys of different length, fields misaligned
- PGP format supports for JPEG picture in key, and picture can be signed
  - JPEG readers ignore garbage after End of Image marker
- Certificate A has RSA-8192 public key, with victim ID
- Certificate B has RSA-6144 public key, and attacker's picture
  - Stuff JPEG in key A, and ID B in JPEG
  - Need very small JPEG: example 181-byte JPEG (*almost compliant*)



# Certificate structure

| | | Key A (RSA-8192) | | Key B (RSA-6144) | |
|---|---|---|---|---|---|
| | 0x0000 | 99 04 0d 04 ** ** ** ** 01 20 00 ?? ?? ?? ?? ?? | | 99 03 0d 04 ** ** ** ** 01 18 00 ?? ?? ?? ?? ?? | ← Prefix |
| | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | |
| | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | |
| | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | |
| | 0x0040 | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | CPC |
| | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | |
| | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | |
| RSA pubkey | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | | ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? | |
| | | ⋮ | | ⋮ | |
| | | Collision here! | | | |
| | 0x0300 | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | !! !! !! !! !! !! !! !! !! !! !! 00 11 01 00 01 | Common Suffix |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | d1 00 00 01 19 c0 57 01 10 00 01 01 00 00 00 00 | |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | 00 00 00 00 00 00 00 00 ff d8 ff db 00 43 00 ff | |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff | |
| JPEG | 0x0340 | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff | |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff | |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff | |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | c0 00 0b 08 00 40 00 58 01 01 11 00 ff c4 00 28 | |
| | 0x0380 | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | 00 01 01 01 00 00 00 00 00 00 00 00 00 00 00 00 | |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | 00 00 04 03 10 01 00 00 00 00 00 00 00 00 00 00 | |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | 00 00 00 00 00 00 ff da 00 08 01 01 00 00 3f 00 | |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | d0 4e a0 01 3a 80 04 ea 01 3a 80 04 e0 00 a0 13 | |
| | 0x03c0 | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | ← | 8a 13 82 84 e2 84 e0 00 00 28 4e 00 0a 13 8a 13 | |
| | | .. .. .. .. .. .. .. .. .. .. .. .. .. !! !! !! | ↔ | a8 00 4e a1 3a 80 4e 28 4e 28 07 ff d9 .. .. .. | |
| | | !! !! !! !! !! !! !! !! !! !! !! !! !! !! !! !! | → | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | |
| | | !! !! !! !! !! !! !! !! !! !! !! !! !! !! !! !! | → | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | |
| | 0x0400 | !! !! !! !! !! !! !! !! !! !! !! 00 11 01 00 01 | → | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | |
| UserID | | b4 00 00 00 19 41 6c 69 63 65 20 3c 61 6c 69 63 | → | .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. | |
| | | 65 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 3e 04 10 | → | .. .. .. .. .. .. .. .. .. .. .. .. .. .. 04 10 | |
| Metadata | | 01 02 00 06 05 02 .. .. .. .. 04 ff 00 00 00 0c | ← | 01 02 00 06 05 02 $$ $$ $$ $$ 04 ff 00 00 00 0c | |

# Impersonation attack

1. Build CP collision with prefixes “99040d04*012000”/“99030d04*011800”
2. Choose JPEG image to include in B, UserID to include in A
3. Select “!!” bytes to make RSA modulus.
4. Ask for a signature of key B.
5. Copy the signature to key A.

- Single chosen-prefix collision can be used to target many victims
- Example keys on https://sha-mbles.github.io
  - Key creation date of our CPC in 2038 to avoid malicious usage

- Reported in May, GnuPG does not trust SHA-1 signatures anymore (CVE-2019-14855)

# Conclusion

SHA-1 signatures can now be abused in practice

- SHA-1 must be deprecated (same attacks as on MD5 in 2007)
  - As long as SHA-1 is supported, downgrade attacks are possible
  - Urgent for SHA-1 signatures
    - SLOTH attack as long as SHA-1 is supported in TLS, SSH [Bhargavan & L., NDSS'16]
    - Rogue CA using SHA-1 X.509 certificates [Stevens & *al.*, C'09]
  - We recommend deprecation everywhere (even HMAC-SHA-1)

```
$ openssl s_client -connect msn.com:443 2>&1 | fgrep 'digest'
Peer signing digest:  SHA1
```

- If you are involved in a project that still supports SHA-1, please take action!

- Side result: breaking 64-bit crypto now costs less than 100 kUS$

# Resources used

- Cluster of 150 nodes / 900 GPUs (GTX 1060)
- 2TB hard drive on master node to store chains for the birthday phase
- External machine with huge RAM for operations in $\mathcal{S}$ (Grid 5000: 1TB, rioc: 3TB)

| Phase | Step | Main resource | Repetitions | Wall time |
|---|---|---|---|---|
| Setup | Preparation of the graph | CPU and RAM | | $\approx$ 1 month |
| Birthday | Computing chains | GPU | | 34 days |
| | Sorting chains | Hard drive | $4\times$ | $\approx$ 1 day |
| | Locating collisions | GPU | $4\times$ | $<$ 1/2 day |
| | Searching in graph | RAM | $4\times$ | $<$ 1/2 day |
| Blocks | Building trail & code | Human Time | $9\times$ | $\approx$ 1 day |
| | Finding intermediate block | GPU | $8\times$ | 3 hours – 3 days |
| | Checking results in graph | RAM | $8\times$ | $<$ 1/2 hour |
| | Finding last block | GPU | $1\times$ | 6 days |

# Current GPU prices

## PRICING

Compare our servers performance and price with major companies such as GPU instances from AWS, GPU instances from google and azure and GPU servers from small competitors. You'll be surprised!

| GPU Instance | GPU RAM | CUDA Cores | Pricing |
|---|---|---|---|
| 6 x GTX 1050 2GB | 12 GB (6 x 2 GB) | 3840 (6 x 640) | $99/mo minimum rental period is 1 month |
| 6 x GTX 1060 3GB | 18 GB (6 x 3 GB) | 6912 (6 x 1152) | $209/mo minimum rental period is 1 month |
| 6 x GTX 1060 6GB | 36 GB (6 x 6 GB) | 7680 (6 x 1280) | $249/mo minimum rental period is 1 month |
| 5 x GTX 1080 8GB | 40 GB (5 x 8 GB) | 12800 (5 x 2560) | $359/mo minimum rental period is 1 month |

**Discounts up to 50% available**

please contact sales for more information

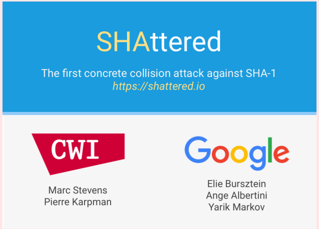
https://www.gpuserversrental.com/

# *SHA-1 Cryptanalysis*

| | | |
|---|---|---|
| *2005-02* | Theoretical collision with $2^{69}$ op. | [Wang & al., Crypto'05] |
| ... | Several unpublished collision attacks in the range $2^{51} - 2^{63}$ | |
| *2010-11* | Theoretical collision with $2^{61}$ op. | [Stevens, EC'13] |
| *2015-10* | Practical freestart collision (on GPU) | [Stevens, Karpman & Peyrin, Crypto'15] |
| *2017-02* | Practical collision with $2^{64.7}$ op. (GPU) | [Stevens & al., Crypto'17] |
| *2019-05* | Theoretical chosen-prefix collision with $2^{67.1}$ op. (GPU) | [L.&P., Eurocrypt'19] |
| *2020-01* | Practical chosen-prefix collision with $2^{63.4}$ op. (GPU) | New! |

## *SHAttered attack: Colliding PDFs*


SHA-1 =
38762cf7f55934b34d17
9ae6a4c80cadccbb7f0a



# *SHA-1 depreciation*

*2006-03* NIST Policy on Hash Functions
Federal agencies should stop using SHA-1 for digital signatures, digital time stamping and other applications that require collision resistance as soon as practical, and must use the SHA-2 family of hash functions for these applications after 2010.

*2011-11* CA/Browser Forum:
"SHA-1 MAY be used until SHA-256 is supported widely by browsers"

*2014-09* CA/Browser Forum depreciation plan

- Stop issuing SHA-1 certificates on 2016-01-01
- Do not trust SHA-1 certificates after 2017-01-01

*2015-10* Browsers consider moving deadline to 2016-07

*2017-0x* Modern browsers reject SHA-1 certificates