Category: EUGridPMA Documents
Status:DRAFT
Document: SHA1Risk-v0.5.doc
Editor: hashrat@eugridpma.org
Last updated: Mon, 25 November 2013
Total number of pages: 7

DRAFT DOCUMENT

subject to revision by the IGTF HASHRAT

Risk Assessment on Hash Function Vulnerabilities

Abstract

The most-commonly used hash algorithm in IGTF PKI implemention today is SHA-1, which is however increasingly vulnerable to attacks and its continued use may soon start posing a threat to the IGTF PKI. However, moving to a more modern hash like SHA-2 (or soon SHA-3) has operational consequences for the e-Infrastructure relying on the IGTF PKI in that not all software implementations can currently work with SHA-2. In this document we assess the risk to attacks on SHA-1 with respect ot the integrity of the trust fabric and the impact of moving to SHA-2 at a given point in time on the operational infrastructure

Table of Contents

1 Scope of this document....2
2 Background....2
2.1 Collision attacks....2
2.2 Chosen-prefix attacks....3
3 Secure Hash Algorithm 2....3
4 Feasibility categorization....3
5 Attack surfaces and impact....3
5.1 Self-signed root certificates....4
5.2 Intermediate and cross-signed CA certificates....4
5.3 End-entity certificates....4
5.3.1 Approaches to the replacement of end entity certificates....5
5.3.2 Other remedial factors....5
5.4 Proxies....5
5.5 CRLs....5
5.6 OCSP....6
5.7 Social engineering and blackmail attacks....6
5.8 RA-CA Communications and request submission....6
5.9 Trust Anchor distribution web site (and TACAR)....6
6 Impact on the infrastructure and remedial actions....6
7 Testing of operating systems and middleware....6
8 Summary and Recommendations....7
9 References....7

1 Scope of this document

The most-commonly used hash algorithm in IGTF PKI implemention today is SHA-1 [FIPS 180-1]. Recommendations by NIST [SP800-107] strongly suggested to move away from SHA-1 by 2010 (ref?) due to the increased vulnerability of SHA-1 in the wake of cryptanalytic attacks and increased computational power making brute-force attacks more feasible. In this document, we try

- to assess the current state of the attacks of SHA-1 as relevant to the IGTF PKI;
- gauge the probability of successfully exploiting such an attacks against the various elements of the PKI infrastructure;
- possible remediations taken by the IGTF to preserve the integrity of the PKI;
- the effect on the operational environment of the various scenarios.

2 Background

SHA-1 was originally considered to have at least 80 collision resistance bits (i.e. finding a collision would take an estimated 2^{80} operations, the square root of 2^{160}). However, the number of useful bits has decreased over the years, to approximately 2^{58} opertations in 2010.

2.1 Collision attacks

This number of operations the above-named collision attacks represents the probable number of attempts needed to generate two messages that hash to the same value, and the new estimate of 2^{58} operations is significantly faster than the inherent vulnerability to the birthday attack [WPedia-SHA1]. This is however still a factor of 2^{34} (~$2x10^{10}$) more complex than the current best attack against MD5, which can be done in about 1 second on a modern-day CPU [Stevens, 2007].
For SHA-1, this translates into ~ 544 CPU-years for finding a SHA-1 collision, an amount of CPU power within reach of most potential attackers.

However, these attacks target *any form of collision* between two plaintexts, and it is far more complex to find a second preimage, that is also a valid message, here an X.509 certificate, which hashes to the same hash as a (different) given X.509 certificate. The number of bits of security of SHA1 as a preimage resistant protocol is 2^{159} [1] which at the above rate would be require 10^{33} CPU years. Nevertheless, it is possible that weaknesses found in collision may introduce weaknesses as a second preimage resistant protocol: it is also possible that a collision itself may aid an attacker: this assumes that the attacker can put seemingly innocuous data into a request and then replace it with more malicious content later. The attack becomes easier if the attacker has a large number of certificates, but not much, because the number of certificates issued by the CAs is much less than the number of keys.

Since X.509 certificates must also parse with existing ASN.1 parsers, it also becomes a lot more difficult if the attacker can only influence a minor portion of the plaintext that is being generated, e.g. only the subject distinguished name in a certificate, or the 'subjectAlternativeName'. These additional complexity factors do not invalidate the attack itself, but do help in mitigating the attack in our community and as such have relevance for the risk assessment.

At the same time, we should keep in mind that the 'natural' roll-over time of certificates in the IGTF is approximately 13-18 months: a few months to update the CA operations, policies and tools, and then up to 13 months validity period for end-entity certificates. Any risk assessment therefore needs to 'project ahead' another 18 months when such an assessment is based on computational feasibility.

[1] The average of the number of attempts required in a brute force attack is half the total number of possibilities.

2.2 Chosen-prefix attacks

The 'chosen prefix' collision attack that finally 'broke' MD5 for PKI applications should not be a threat to SHA-1, since that type of attack is specific to Merkle–Damgård hash functions [WPedia-Collision].

3 Secure Hash Algorithm 2

The Secure Hash Algorithms 2 is a family of four cryptographic hash functions: SHA-224, SHA-256, SHA-384, and SHA-512 [FIPS180-4].
Of these four functions, SHA-224 and SHA-384 are reduced versions of the other two (SHA-256 and SHA-512, respectively), and their calculation actually implies calculating the full 256 or 512 bit hash function with different initialization vectors and omitting part of the hash ourput. They exist primarily to reduce message lengths, but since message length is not a primary concern in this case, they should not be used. Moreover, implementations of cryptographic libraries (particularly in earlier <= 1.6 version of Java) do not correctly calculate or process the shorter SHA-224 hash.

For all IGTF purposes, of the SHA-2 family only SHA-256 and SHA-512 must be used.

4 Feasibility categorization

The risk to the infrastructure depends on the feasibility of actually generating fake X.509 certificates. Assuming the amount of CPU power needed to find a functional but false X.509 certificate is non-trivial, the risk depends on the amount of CPU an attacker can harness. For practical purposes we categorize feasibility in four groups

1. the attack is trivially feasible by anyone with a modern computer (~ 1-10 CPU days);
2. the attack is readily possible, and can be run on an infrastructure many people can harness, e.g. the e-Infrastructure's own resources available to a random systems administrator in a site (~ 500 000 CPU-days), since you will be able to run undetected and unchecked for a few hours, but it will also take a long time to get banned on all sites;
3. the attack requires determination, and an attacker will need to collect enough other credentials and/or be able and willing to rent large bot nets in order to successfully generate a usable collision;
4. the attack is complex and more costly than the value of the infrastructure which is protected by the IGTF PKI, i.e. only very resourceful attackers will be able to exploit it (governmental types, large corporations).

The 'feasibility' should be assessed keeping in mind the 13-18 months life cycle of valid end-entity certificates.

5 Attack surfaces and impact

Depending on the attack surface (which parts of the PKI are vulnerable to a particular attack), and the feasibility of such an attack, remedial measures have to be taken. Possible actions could include

- the revocation and re-issuance of all end-entity certificates;
- revocation of intermediate CAs (and redistribution of all trust anchors)
- (suggest to) modify software to reject any SHA-1 hashes in path validation
- inspection of all incoming applications for 'suspected rogue' data to be embedded in the issued certificate (pre-screening)
- ensuring randomness in the issued end-entity certificates (to prevent 'prediction' attacks against CAs with a low issuance volume), e.g. by using random serial numbers, adding random data in non-critical or comment extensions, etc.

All CAs MUST be able to issue SHA-2, and be prepared to so do on short notice.
The conversion also must start as soon as a 'one-of-a-kind' cryptanalysis attack has succeded, in which case the migration should start immediately.

5.1 Self-signed root certificates

Self-signed root certificates are not affected by any hash weaknesses, since these are installed out-of-band in a local trust anchor repository. In such a local installation, the signature on the public key and other certificate data does not provide any security and is present only for validation consistency's sake.

5.2 Intermediate and cross-signed CA certificates

In the case of intermediate CA certificates and any cross-signatures, all input data to the X.509 structure is generated by trustworthy parties. None of the original plaintext can be chosen by the attacker, who will have to find a collision based only on an already fixed hash value.

However, an attack may be able to convert an end-entity certificate: while it would normally require a second preimage attach, a collision could possibly be exploited by an attacker if they, rather than using an existing end-entity certificate, put material into a certificate request which looks natural to the CA, and then replace it with other material later which the CA would not have approved. The end-entity certificate could then be converted into an intermediate or cross-signed CA certificate, e.g. by being able to alter the 'basicConstraints' extension, or by removing that extension altogether. In case such an attack is found, any non-self-signed certificate is exposed.

Although intermediate (and cross-signed) CA certificates are typically installed in a trust anchors store in the IGTF PKI environment, an attack may include sending any such intermediate certificates as part of an SSL handshake, in which case the 'attacker-provided' intermediate CA certificates will be used in path validation up to (but *not* including) a self-signed root. This potential attack route will thus affect most non-grid software.

5.3 End-entity certificates

End-entity certificates contain (a limited amount of) contributed data such as subject distinguished name, subject alternative names, or other elements. How the rest of the data therein is generated depends on the issuing CA – there may be more data that can be manipulated (even if not influenced directly) by the applicant-attacker. For example, if extensions can be embedded in the request and are honored in the issued certificate (without further validation), there could be the option to insert random data into the issued certificate, thereby facilitating hash collision attacks.

Since the attack is directed against the algorithm and not any specific EEC,

Feasibility	Remediation and time line for moving to SHA-2
trivial	stop using the SHA-1 algorithm, and move to SHA-2 immediately. Unless SHA-2 is not available to the CA, in which case operations should be stopped completely. This also includes stopping accepting SHA-1 CSRs. All requestors will have to reapply using the original validation process, and *not* allow for renewal based on any original SHA-1 based credentials. CAs that fail to issue a SHA-2 based CRL in time, must be removed from the distribution and the distribution re-issued.
readily possible	CA has to be ready to issue SHA-2. Then the CA can re-issue based on the existing key material *and* revoke all the SHA-1 based serials. The CRL should then also be SHA-2, since that will disable software that cannot process SHA-2.

	This saves the subscriber from re-doing the F2F. *If there is a risk of back-compromise and the attacker can get access to the original subscribers private key, renewal will not help and re-keying is necessary.*
determination	
costly and complex	Execute migration on scheduled date

5.3.1 Approaches to the replacement of end entity certificates

5.3.1.1 Reissuance of certificates

Reissuance of end-entity certificates based on existing key material: process would require notifying users, have them download the new cert, pair it up with the private key, and replace the previous certificates. Browsers are not good at supporting this process: if a certificate is deleted, the private key is deleted with it. If a new certificate is downloaded, it will not necessarily

5.3.1.2 Natural rekey process

Having users rekey their existing certificates (client software which does this)

5.3.1.3 Asking users to rekey

This method has important drawbacks in that it requires users to take active action (which a proportion of the users will miss out on, necessitating a re-vetting process), and in addition may have the effect of having a mass issuance on a specific date, which for classic CAs will cause problems when the routine re-keying is due one year later.

5.3.2 Other remedial factors

The following elements and options should be considered

- Can certificates be monitored to prevent the exploitation of a collision? E.g. the length of the request, or a stricter syntax checking of strings and extensions?
- Extensions in requests SHOULD be ignored – does all CA software ignore them?
- Can the middleware do more to protect itself, e.g. by actively disabling certain hash algorithms?

5.4 Proxies

Given the relatively short length of proxy RSA keys (some are still 512 bits by default), and the fact that these are stored without additional protections or with only operating-system level permissions on timeshared systems, the typical SHA-1 vulnerabilities will usually be more complex than other attacks against proxy certs.

For consistency's sake, it is advisable to have the software that parses the EEC and proxy certificates to have the same set of supported hash algorithms.

5.5 CRLs

Keep in mind that also the ability to publish functional CRLs can be impaired if these are still signed with SHA-1, or if the serial number of 'rogue' certificates can be altered at-will – since revocation is based on the issuer name and serial number.

5.6 OCSP

Can OCSP (On-line Certificate Status Protocol) responses be fakes or modified, allowing either a denial of service or inappropriate validation of certificates?

5.7 Social engineering and blackmail attacks

Including denial-of-service by threat, even if there is no real vulnerability but a significant enough perceived risk due to the algorithms you use.

5.8 RA-CA Communications and request submission

Can the process of conveing request and vetting information (typically internal to a CA) be attacked because it relies on an insecure hash? This question is relevant both for CA-RA communication that is conducted via secured email, as well as on-line processes (via secure LDAP, signed SAML statements, or similar) where a digest is used to secure the message.

5.9 Trust Anchor distribution web site (and TACAR)

There should be multiple integrity checking mechanisms available for the same set of trust anchors. Currently, the Debian packaging provides for a SHA-2 based check, besides MD5 and SHA-1 which are also provided.

Feasibility	Remediation and time line for moving to SHA-2
trivial	Becomes 'useless' unless the SSL server cert is SHA-2, except for the Debian packaging where several hash functions are used simultaneously (making the attack extremely hard) and where SHA-256 is already used by default.

Multiple checksums may also be provided in non-automated ways, e.g. through web sites for checking. The checksums should preferably be from different families.

6 Impact on the infrastructure and remedial actions

The impact of changing to a more modern hash algorithm, such as SHA-2, on the operational infrastructure has been described in [Litmaath, 2012], also in conjunction with a concurrent move to the exclusive use of RFC3820 proxies. An move to SHA-2 in 2012 would result in significant disruptions to the infrastructure which should be weighed against the risk of continued use of SHA-1 in the IGTF PKI.

The range of infrastructures that rely on the IGTF is large and broad, and all employ multiple and distinct software suites implementing PKI functions. These infrastructures include for example EGI, OSG, PRACE, PRAGMA, and XSEDE, as well as all domain-specific infrastructures like wLCG, and national or regional infrastructures. This risk assessment primarily targets those infrastructures directly or indirectly represented in the IGTF.

7 Testing of operating systems and middleware

The following issues should be addressed in a testing document, which is to be provided and populated by the software providers and infrastructure operators.
It is noted that not all Safenet eToken (PRO) tokens support SHA-2, only the most recent versions do. These tokens are often used for holding Robot certificate key pairs for those CAs that rely on hardwate tokens for such certificates.

8 Summary and Recommendations

The latest set of recommendations and the migration time line is communicated via separate channels. This risk assessment will be refined on a continuous basis taking into account both the vulnerability of the hash algorithm as well as the state of deployed software and the readiness of the operational infrastructure.

The IGTF, in coordination with the operations and deployment experts of our major relying parties, has agreed on an implementation time line for SHA-2. The cut-off dates associated with the different stepts are maintained by the EUGridPMA at

http://www.eugridpma.org/documentation/hashrat/sha2-timeline

It should be notes that selected authorities MAY continue to use the SHA-1 digest in intermediate and end-entity certificate for specific operational reasons, but the risk of compromise is then entirely borne by the authority: in case of 'compromise' of the SHA-1 mechanism these authorities will be suspended from the IGTF distribution forthwith, commensurate with maintaining the integrity and trustworthiness of the IGTF trust fabric.

9 References

[FIPS 180-1] *Secure Hash Standard*, United States of America, National Institute of Science and Technology, Federal Information Processing Standard (FIPS) 180-1, April 1993.

[FIPS180-4] *Secure Hash Standard*, United States of America, National Institute of Science and Technology, Federal Information Processing Standard (FIPS) 180-4, March 2012.

[WPedia-SHA1] http://en.wikipedia.org/wiki/SHA-1#Cryptanalysis_and_validation

[Stevens, 2007] M.M.J. Stevens. *On Collisions for MD5*. (June 2007) "[...] we are able to find collisions for MD5 in about $2^{24.1}$ compressions for recommended IHVs which takes approx. 6 seconds on a 2.6GHz Pentium 4.". http://www.win.tue.nl/hashclash/On Collisions for MD5 - M.M.J. Stevens.pdf

[WPedia-Collision] http://en.wikipedia.org/wiki/Collision_attack#Chosen_prefix_collision_attack

[Litmaath, 2012] Maarten Lidmaath, CERN, "RFC Proxies", wLCG GDB 11 January 2012: http://indico.cern.ch/materialDisplay.py?sessionId=4&materialId=0&confId=155064