Cryptographers Demonstrate Collision in Popular SHA-1 Algorithm

TECHNICAL ANALYSIS BY BILL BUDINGTON | FEBRUARY 24, 2017

On February 23rd, a joint team from the CWI Amsterdam and Google announced that they had generated the first ever collision in the SHA-1 cryptographic hashing algorithm. SHA-1 has long been considered theoretically insecure by cryptanalysts due to weaknesses in the algorithm design, but this marks the first time researchers were actually able to demonstrate a real-world example of the insecurity. In addition to being a powerful Proof of Concept (POC), the computing power that went into generating the proof was notable:

> We then leveraged Google's technical expertise and cloud infrastructure to compute the collision which is one of the largest computations ever completed.
>
> Here are some numbers that give a sense of how large scale this computation was:
>
> - Nine quintillion (9,223,372,036,854,775,808) SHA1 computations in total
> - 6,500 years of CPU computation to complete the attack first phase
> - 110 years of GPU computation to complete the second phase

The CWI Amsterdam and Google researchers launched shattered.io, a site explaining the attack and linking to two distinct pdf files: shattered-1.pdf and shattered-2.pdf with different contents but the same SHA-1 checksum.

What is SHA-1, anyway?

SHA-1 is part of a class of algorithms known as collision-resistant hashing functions, which create a short digest of some arbitrary data. That can be a piece of text, a database entry, or a file, just to name a few examples. For instance, the SHA-1 result or 'checksum' of the first sentence in this paragraph is `472825ab28b45d64cd234a22398bba755dd56485`. Creating a digest of data is useful in many contexts. For example, making a cryptographic signature for a digest is more convenient and faster than signing the entire contents of a message, a fact that many cryptographic systems have taken advantage of. Lots of software uses this type of hashing function, and relies on the collision-resistance property to verify that the contents of the original message haven't been corrupted or tampered with.

Sunsetting SHA-1

While a brute-force attack (simply trying all the possibilities until a collision is found) remains impractical, low-level analysis of the algorithm has revealed deep fractures in its design. Over time, as these theoretical attacks against the algorithm have gotten better, many have moved away from SHA-1 to guarantee security. In 2014, the CA Browser Forum (an organization which comprises the trust-roots for the web) passed a ballot which prevented new HTTPS certificates from being issued using SHA-1 after 2015. And earlier this year, the major browsers started to remove support for HTTPS sites which serve SHA-1 certificates. In general, companies and software projects were moving away from relying on SHA-1. Next-generation hashing algorithms such as SHA-256 and SHA-3 have been available for a long time, and provide far better guarantees against collisions.

So what's the big deal?

Unfortunately, the migration away from SHA-1 has not been universal. Some programs, such as the version control system Git, have SHA-1 hard-baked into its code. This makes it difficult for projects which rely on Git to ditch the algorithm altogether. The encrypted e-mail system PGP also relies on it in certain places.

While initially promising to deprecate SHA-1 in a similar time-frame as the other browsers, Internet Explorer has pushed that date back to mid-2017. This means that sites with certificates signed by the insecure function will still be trusted for IE users. And while the collision was demonstrated on two pdf files, there is nothing stopping others from crafting a malicious X.509 certificate with the same checksum as a valid certificate, and using that to impersonate a legitimate HTTPS site. History (and Moore's law) shows us that this only becomes easier over time. The first full collision of the then-popular MD5 hashing algorithm was demonstrated in August 2004. Less than seven months later, an X.509 collision was shown.

Last year, we pointed out that a SHA-1 collision in 2017 was entirely foreseeable, and will happen again in the future. To have robust protections against cryptographic vulnerabilities, software projects have to take these vulnerabilities seriously *before* they turn into demonstrated attacks, when they are still theoretical but within the realm of possibility. Otherwise, the time it takes to migrate away from these insecure algorithms will be well used by attackers, as well.

SECURITY SECURITY ENCRYPTION

JOIN EFF LISTS