CipherShed - Bug #108

Open Crypto Audit Project TrueCrypt CS-TC-2 - AES implementation susceptible to cache-timing attacks

04/05/2015 02:42 AM - Jason Pyeron

Status:	New	**Start date:**	04/05/2015
Priority:	Urgent	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour

Description

TARGETS: The AES implementations in AesSmall.c, AesSmall_x86.asm, Aes_x86.asm, Aes_x64.asm
DESCRIPTION: Naive optimizations of AES depends heavily on large look-up tables. The indices into
these tables depend both on attacker-supplied plaintext as well as secret key material. Because the
tables are so large, they do not comfortably fit inside the data cache of some CPUs. By choosing inputs
carefully, an attacker can induce variable timing dependent on secret key material. By measuring
these timings and making statistical inferences, they can recover secret keys completely. For more
information, see Cache-timing attacks on AES by Daniel J. Bernstein [http://cr.yp.to/papers.html#cachetiming] and Efficient Cache Attacks on
AES, and Countermeasures by Eran Tromer et al. [http://www.tau.ac.il/~tromer/papers/cache-joc-20090619.pdf]
TrueCrypt provides multiple implementations of AES. Many of these depend on look-up tables, including
those in AesSmall.c, AesSmall_x86.asm, Aes_x86.asm, and Aes_x64.asm. The C implementation
by Brian Gladman in AesSmall.c is known to be vulnerable to cache-timing attacks.[http://cr.yp.to/mac/variability1.html], [http://www.metzdowd.com/pipermail/cryptography/2005-June/008946.html]
Also provided is an assembler implementation using the Intel AES-NI hardware instructions. Implementations
using these instructions do not rely on the data cache and are not vulnerable to cachetiming
instructions. This is supported by several research papers.[Are AES x86 Cache Timing Attacks Still Feasible?, Keaton Mowery et al.], [Fine grain Cross-VM Attacks on Xen and VMware are possible!, Gorka Irazoqui Apecechea et al.]
The AES-NI implementation is preferred on platforms where the requisite instructions are available.
EXPLOIT SCENARIO: An attacker may be able to extract AES keys used to protect encrypted volumes.
A successful exploit may rely on the attacker's ability to execute native code on the victim's machine,
but recent advances in cache attacks performed by untrusted JavaScript [http://arxiv.org/abs/1502.07373v1] indicate this area is being
researched more heavily.
Recommendation: Writing high-performance, constant-time, portable software implementations of
AES is a difficult undertaking. A non-portable Intel 64-bit implementation of AES-CTR by Käsper is
available.[https://cryptojedi.org/crypto/index.shtml#aesbs] Besides this implementation, it is possible to partially mitigate the vulnerability to cachetiming
side channels. Some strategies include:
• Oblivious table look-ups that scan the entire look-up table, selecting the value desired on the
way through.
• Implementing compressed tables to protect the outer two rounds only. While it is still possible
to attack the inner rounds, the attack complexity grows considerably.