expat SOS Fund Audit Fix Log

Identified Vulnerabilities

MOX-001 Randomized hash function vulnerable to collisions, causing worst-case run-time performance (Moderate)

Fixes: https://github.com/libexpat/libexpat/pull/39/commits

VERIFIED

MOX-002 Huge input can cause an application crash on 32-bit systems (Moderate)

Fixes:
https://github.com/libexpat/libexpat/commit/d4f735b88d9932bd5039df2335eefdd0723dbe20
https://github.com/libexpat/libexpat/commit/70db8d2538a10f4c022655d6895e4c3e78692e7f
https://github.com/libexpat/libexpat/commit/4be2cb5afcc018d996f34bbbce6374b7befad47f
https://github.com/libexpat/libexpat/commit/7e5b71b748491b6e459e5c9a1d090820f94544d8

Sebastian Pipping (expat): This is relevant where XML_CONTENT_BYTES is *not* defined. Default is defined to 1024 bytes.

VERIFIED

MOX-003 xmlparse.c uses uninitialized memory 'next' in function processInternalEntity (Moderate)

Fix:
https://github.com/libexpat/libexpat/commit/a4dc944f37b664a3ca7199c624a98ee37babdb4b

Sebastian Pipping (expat): This was addressed with MOX-004.

VERIFIED

MOX-004 xmlparse.c uses uninitialized memory 'next' in function internalEntityProcessor (Moderate)

Fix:
https://github.com/libexpat/libexpat/commit/a4dc944f37b664a3ca7199c624a98ee37babdb4b

Sebastian Pipping (expat): This was addressed with MOX-003.

VERIFIED

MOX-005 Salt generation for the randomized hash is weak and could leak addresses (Low)

Fixes: https://github.com/libexpat/libexpat/pull/30/commits

VERIFIED

MOX-006 In xmlparse.c, several XML_Set* and XML_Get* APIs take a pointer argument but do not check for NULL before dereferencing (Low)

Fixes:
https://github.com/libexpat/libexpat/commit/d37f74b2b7149a3a95a680c4c4cd2a451a51d60a
https://github.com/libexpat/libexpat/commit/9ed727064b675b7180c98cb3d4f75efba6966681
https://github.com/libexpat/libexpat/commit/6a747c837c50114dfa413994e07c0ba477be4534

VERIFIED

MOX-007 The XML_FreeContentModel, XML_MemFree and XML_MemRealloc APIs do not (or have no way to) check user supplied memory pointers (Low)

Sebastian Pipping (expat): To check user-supplied pointers, one could: Allocate some more bytes at allocation time, put some magics bytes in for identification, and check for presence of these bytes when freeing or when reallocating. The question is what to do when a foreign or previously freed pointer is detected. Options include: (a) not freeing the pointer silently and going on or (b) terminating the process. For (a) the implication is that it's safer to continue than to terminate. However, that is impossible to know for every scenario. For (b) at least with glibc freeing the same pointer twice already terminates the process. In general, it needs to be asked why the memory allocation of Expat needs hardening that allocations in the rest of the application outside of Expat do not need. If the application uses safety wrappers to malloc/realloc/free already, these can be passed to Expat to make it on par with the rest of the application. Expat would not change in that case. To summarize, there are no plans to add pointer validation at the moment.

Mahesh Saptarshi (Radically Open Security): *Deferring to Sebastian's views - successful exploit of libexpat would require exploitable application linking with libexpat.*

VERIFIED