REFERENCES

[1] Sam Ainsworth and Timothy M Jones. 2020. MarkUs: Drop-in use-after-free prevention for low-level languages. In *Proceedings of the 41st IEEE Symposium on Security and Privacy (Oakland)*. San Francisco, CA.

[2] Thanassis Avgerinos, Sang Kil Cha, Alexandre Rebert, Edward J Schwartz, Maverick Woo, and David Brumley. 2011. AEG: Automatic exploit generation. In *Proceedings of the 18th Annual Network and Distributed System Security Symposium (NDSS)*. San Diego, CA.

[3] blackngel. 2009. Malloc Des-Maleficarum. http://phrack.org/issues/66/10.html.

[4] David Brumley, Pongsin Poosankam, Dawn Song, and Jiang Zheng. 2008. Automatic patch-based exploit generation is possible: Techniques and implications. In *Proceedings of the 29th IEEE Symposium on Security and Privacy (Oakland)*. Oakland, CA.

[5] Silvio Cesare. 2020. Breaking Secure Checksums in the Scudo Allocator. https://blog.infosectcbr.com.au/2020/04/breaking-secure-checksums-in-scudo_8.html.

[6] Sang Kil Cha, Thanassis Avgerinos, Alexandre Rebert, and David Brumley. 2012. Unleashing mayhem on binary code. In *Proceedings of the 33rd IEEE Symposium on Security and Privacy (Oakland)*. San Francisco, CA.

[7] Wei Chan. 2019. Heap Overflow Exploitation on Windows 10 Explained.

[8] Weiteng Chen, Xiaochen Zou, Guoren Li, and Zhiyun Qian. 2020. KOOBE: Towards Facilitating Exploit Generation of Kernel Out-Of-Bounds Write Vulnerabilities. In *Proceedings of the 29th USENIX Security Symposium (Security)*. Boston, MA.

[9] Jong-Deok Choi and Andreas Zeller. 2007. Isolating failure-inducing thread schedules. In *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)*.

[10] Moritz Eckert, Antonio Bianchi, Ruoyu Wang, Yan Shoshitaishvili, Christopher Kruegel, and Giovanni Vigna. 2018. Heaphopper: Bringing bounded model checking to heap implementation security. In *Proceedings of the 27th USENIX Security Symposium (Security)*. Baltimore, MD.

[11] Google. 2013. Partition Alloc Design. https://chromium.googlesource.com/chromium/src/+/master/base/allocator/partition_allocator/PartitionAlloc.md.

[12] GrapheneOS. 2018. Hardened malloc. https://github.com/GrapheneOS/hardened_malloc.

[13] Alex Groce, Mohammed Amin Alipour, Chaoqiang Zhang, Yang Chen, and John Regehr. 2014. Cause reduction for quick testing. In *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)*. San Jose, CA.

[14] Alex Groce, Mohammad Amin Alipour, Chaoqiang Zhang, Yang Chen, and John Regehr. 2016. Cause reduction: delta debugging, even without bugs. *Software Testing, Verification and Reliability* 26, 1 (2016), 40–68.

[15] Mouna Hammoudi, Brian Burg, Gigon Bae, and Gregg Rothermel. 2015. On the use of delta debugging to reduce recordings and facilitate debugging of web applications. In *Proceedings of the 10th European Software Engineering Conference (ESEC) and ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE)*. Bergamo, Italy.

[16] Ben Hawkes. 2019. 0day "In the Wild". https://googleprojectzero.blogspot.com/p/0day.html.

[17] Sean Heelan, Tom Melham, and Daniel Kroening. 2018. Automatic Heap Layout Manipulation for Exploitation. In *Proceedings of the 27th USENIX Security Symposium (Security)*. Baltimore, MD.

[18] Sean Heelan, Tom Melham, and Daniel Kroening. 2019. Gollum: Modular and Greybox Exploit Generation for Heap Overflows in Interpreters. In *Proceedings of the 26th ACM Conference on Computer and Communications Security (CCS)*. London, UK.

[19] Lukas Kirschner, Ezekiel Soremekun, and Andreas Zeller. 2020. Debugging inputs. In *Proceedings of the 42nd International Conference on Software Engineering (ICSE)*.

[20] Byoungyoung Lee, Chengyu Song, Yeongjin Jang, Tielei Wang, Taesoo Kim, Long Lu, and Wenke Lee. 2015. Preventing Use-after-free with Dangling Pointers Nullification. In *Proceedings of the 2015 Annual Network and Distributed System Security Symposium (NDSS)*. San Diego, CA.

[21] Beichen Liu, Pierre Olivier, and Binoy Ravindran. 2019. SlimGuard: A Secure and Memory-Efficient Heap Allocator. In *Proceedings of the 20th International Middleware Conference*.

[22] LLVM Project. 2019. Scudo Hardened Allocator. https://llvm.org/docs/ScudoHardenedAllocator.html.

[23] Kangjie Lu, Marie-Therese Walter, David Pfaff, Stefan Nürnberger, Wenke Lee, and Michael Backes. 2017. Unleashing use-before-initialization vulnerabilities in the Linux kernel using targeted stack spraying. In *Proceedings of the 2017 Annual Network and Distributed System Security Symposium (NDSS)*. San Diego, CA.

[24] Microsoft. 2018. Low-fragmentation Heap. https://docs.microsoft.com/en-us/windows/win32/memory/low-fragmentation-heap.

[25] Microsoft. 2019. mimalloc. https://github.com/microsoft/mimalloc.

[26] Matt Miller. 2020. Pursuing Durably Safe Systems Software. In *SSTIC*.

[27] Ghassan Misherghi and Zhendong Su. 2006. HDD: hierarchical delta debugging. In *Proceedings of the 28th International Conference on Software Engineering (ICSE)*. Shanghai, China.

[28] Gene Novark and Emery D Berger. 2010. DieHarder: securing the heap. In *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS)*. Chicago, IL.

[29] Angelos Oikonomopoulos, Elias Athanasopoulos, Herbert Bos, and Cristiano Giuffrida. 2016. Poking holes in information hiding. In *Proceedings of the 25th USENIX Security Symposium (Security)*. Austin, TX.

[30] Rohan Padhye, Caroline Lemieux, Koushik Sen, Mike Papadakis, and Yves Le Traon. 2019. Semantic fuzzing with zest. In *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)*. Beijing, China.

[31] John Regehr, Yang Chen, Pascal Cuoq, Eric Eide, Chucky Ellison, and Xuejun Yang. 2012. Test-case reduction for C compiler bugs. In *Proceedings of the 2012 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*. Beijing, China.

[32] Dusan Repel, Johannes Kinder, and Lorenzo Cavallaro. 2017. Modular Synthesis of Heap Exploits. In *Proceedings of the ACM SIGSAC Workshop on Programming Languages and Analysis for Security*. Dallas, TX.

[33] Chris Rohlf. 2020. https://github.com/struct/isoalloc.

[34] Edward J Schwartz, Thanassis Avgerinos, and David Brumley. 2011. Q: Exploit Hardening Made Easy.. In *Proceedings of the 20th USENIX Security Symposium (Security)*. San Francisco, CA.

[35] Konstantin Serebryany, Derek Bruening, Alexander Potapenko, and Dmitriy Vyukov. 2012. AddressSanitizer: A fast address sanity checker. In *Proceedings of the 2012 USENIX Annual Technical Conference (ATC)*. Boston, MA.

[36] shellphish. 2016. how2heap: A repository for learning various heap exploitation techniques. https://github.com/shellphish/how2heap.

[37] Sam Silvestro, Hongyu Liu, Corey Crosser, Zhiqiang Lin, and Tongping Liu. 2017. Freeguard: A faster secure heap allocator. In *Proceedings of the 24th ACM Conference on Computer and Communications Security (CCS)*. Dallas, TX.

[38] Sam Silvestro, Hongyu Liu, Tianyi Liu, Zhiqiang Lin, and Tongping Liu. 2018. Guarder: A tunable secure allocator. In *Proceedings of the 27th USENIX Security Symposium (Security)*. Baltimore, MD.

[39] Michael Steranka and Emery Berger. 2019. Entroprise. https://github.com/plasma-umass/entroprise.

[40] Student. 1908. The probable error of a mean. *Biometrika* (1908), 1–25.

[41] Chengnian Sun, Yuanbo Li, Qirun Zhang, Tianxiao Gu, and Zhendong Su. 2018. Perses: Syntax-guided program reduction. In *Proceedings of the 40th International Conference on Software Engineering (ICSE)*. Gothenburg, Sweden.

[42] Erik Van Der Kouwe, Vinod Nigade, and Cristiano Giuffrida. 2017. Dangsan: Scalable use-after-free detection. In *Proceedings of the 12th European Conference on Computer Systems (EuroSys)*. Belgrade, RS.

[43] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. 2020. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature methods* 17, 3 (2020), 261–272.

[44] Yan Wang, Chao Zhang, Xiaobo Xiang, Zixuan Zhao, Wenjie Li, Xiaorui Gong, Bingchang Liu, Kaixiang Chen, and Wei Zou. 2018. Revery: From Proof-of-Concept to Exploitable. In *Proceedings of the 25th ACM Conference on Computer and Communications Security (CCS)*. Toronto, ON, Canada.

[45] Brian Wickman, Hong Hu, Insu Yun Daehee Jang, JungWon Lim Sanidhya Kashyap, and Taesoo Kim. 2021. Preventing Use-After-Free Attacks with Fast Forward Allocation. (Aug. 2021).

[46] Wei Wu, Yueqi Chen, Xinyu Xing, and Wei Zou. 2019. KEPLER: Facilitating Control-flow Hijacking Primitive Evaluation for Linux Kernel Vulnerabilities. In *Proceedings of the 28th USENIX Security Symposium (Security)*. Santa Clara, CA.

[47] Wei Wu, Yueqi Chen, Jun Xu, Xinyu Xing, Xiaorui Gong, and Wei Zou. 2018. FUZE: Towards facilitating exploit generation for kernel use-after-free vulnerabilities. In *Proceedings of the 27th USENIX Security Symposium (Security)*. Baltimore, MD.

[48] Insu Yun. 2021. mimalloc issue # 372. https://github.com/microsoft/mimalloc/issues/372.

[49] Insu Yun, Dhaval Kapil, and Taesoo Kim. 2020. Automatic Techniques to Systematically Discover New Heap Exploitation Primitives. In *Proceedings of the 29th USENIX Security Symposium (Security)*. Boston, MA.

[50] Michal Zalewski. 2014. american fuzzy lop. http://lcamtuf.coredump.cx/afl/.

[51] Andreas Zeller. 1999. Yesterday, my program worked. Today, it does not. Why?. In *Proceedings of the 7th European Software Engineering Conference (ESEC) / 7th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE)*. Toulouse, France.