Heaven or Hell: Vulnerability Defense Characterization and Evolution

Peter Ullrich

Heaven or Hell: Vulnerability Defense Characterization and Evolution

Peter J. Ullrich
University of Groningen

Abstract

Memory vulnerabilities and bugs become one of the most severe problem in today's system security. Due to the low-level languages' unsound memory control and protection, an increasing number of memory vulnerabilities exist over years. Researchers and practitioners propose many defense mechanisms, trying to overcome the shortcomings of the current memory protection model. We provide a survey on these mechanisms and compare them against each other. We discuss the strengths and weaknesses of each approach. We also show why most of these methods are not adopted by today's commodity operating systems, and offer suggestions on how to find a sweet spot between usability and security.

1 Vulnerabilities and Defenses

Onze eisen voor *Precision* en *Object Awareness* zijn ontworpen om ruimtelijke en temporele geheugenveiligheid te handhaven, die we hier definieren en vervolgens gebruiken om het begrip van een capaciteits-ID te introduceren.

Spatial Vulnerabilities: ook bekend als grensoverschrijdende beveiligingsschendingen - Zijn over- of onderstromen van een object. Over/onderstromen komen voor wanneer een aanwijzer is verhoogd/verlaagd buiten de grenzen van het object dat het momenteel is geassocieerd met. Zelfs als de out-of-bound pointer nog steeds wijst op een geldige object [15, 16, 3], het heeft geen geschiktheid voor het verwijzende object en de operatie resulteert in een schending van ruimtelijke geheugen veiligheid [2, 6, 4, 8]. Echter, deze overtreding is *only* geactiveerd op een verwijzing van een buitenbalkwijzer. De C standaard staat specifiek toe dat er geen wijzigingen beschikbaar zijn [10, 1].

Temporal Vulnerabilities: ook bekend als levenslange veiligheids overtredingen - optreden wanneer het object dat de aanwijzer van een aanwijzer verwijst niet meer is toegewezen en die aanwijzer wordt genegeerd. Voor stapelobjecten is dit omdat het stapelframe van het object is niet langer geldig (de functie waarin het is gemaakt teruggegeven); Voor heapobjecten gebeurt dit als gevolg van een gratis. Deze fouten veroorzaken *necessarily* segmenteringsfouten (toegang tot unmapped memory), omdat het geheugen kan hebben opnieuw toegewezen aan een nieuw object. Op dezelfde manier kunnen we niet alleen opsporen welke geheugen er momenteel is toegewezen, omdat het object op een bepaald adres kan veranderen, wat nog steeds een tijdelijke veiligheidsschending veroorzaakt. Temporale bugs zijn in de hart van veel recente uitbuiting, bijvoorbeeld voor Google Chrome of Mozilla Firefox als getoond in de pwn2own wedstrijden [13]. ze bestaan uit veel defensiemechanismen. ASLR [14, 12, 5] is een zeer belangrijke die de mogelijkheden van aanvallers aanzienlijk kan verzwakken.

Schending van elk type geheugenveiligheid kan worden geformuleerd als een mogelijkheid schending [9]. In onze terminologie is een object een discreet geheugengebied, gemaakt door een toewijzing ongeacht de locatie (stapel, hoop, data, bss onder de Linux ELF formaat). Een mogelijkheid identificeert een specifiek object, samen met informatie over zijn grenzen en toewijzingsstatus [11, 7]. Pointers behouden een capaciteit ID dat identificeert de mogelijkheid van het meest recente object toegewezen - hetzij direct uit de toewijzing of indirect door aliasing een andere aanwijzer. Capaciteiten vormen een contract, op dereference: (i) de aanwijzer moet in de grenzen zijn en (ii) het genoteerde object moet nog steeds worden toegewezen. Schending van de voorwaarden van dit contract leidt tot ruimtelijk of respectievelijk tijdelijke geheugen veiligheidsfouten.

2 Conclusion

We beoordelen geheugensveiligheidsmethoden die volledig voorzien zijn bescherming van gebruikersruimte, inclusief libc, en sterke probabilistische

temporale bescherming. Het is het eerste dergelijke mechanisme dat voldoet aan alle eisen voor een volledige geheugenveiligheidsoplossing, waarbij slechts een bescheiden prestatie optreedt overhead in vergelijking met de state-of-the-art. Is precies, object bewust, uitgebreid in de dekking, en precies. Wij beschermen alle gebruikersruimte volledig geheugen, inclusief de stapel, die ondanks de grootste bron van aanwijzingen is, bleef grotendeels onbeschermd. Tenslotte produceren we nul-foute negatieven en nul authentieke valse positieven in de voorbeeldpakket kwetsbaarheid, wat een belangrijke vooruitgang vertegenwoordigt in de bestaande geheugenveiligheid mechanismen.

References

[1] BACKES, M., BUGIEL, S., AND DERR, E. Reliable third-party library detection in android and its security applications. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (2016), ACM, pp. 356–367.

[2] BACKES, M., HOLZ, T., KOLLENDA, B., KOPPE, P., NÜRNBERGER, S., AND PEWNY, J. You can run but you can't read: Preventing disclosure exploits in executable code. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (2014), CCS '14.

[3] CARBONE, M., CUI, W., LU, L., LEE, W., PEINADO, M., AND JIANG, X. Mapping Kernel Objects to Enable Systematic Integrity Checking. In *Proceedings of the 16th ACM Conference on Computer and Communications Security* (November 2009).

[4] CHEN, Y., KHANDAKER, M., AND WANG, Z. Pinpointing vulnerabilities. In *Proceedings of the 2017 ACM Asia Conference on Computer and Communications Security* (2017), ACM, pp. 334–345.

[5] CHEN, Y., WANG, Z., WHALLEY, D., AND LU, L. Remix: On-demand live randomization. In *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy* (2016), ACM, pp. 50–61.

[6] CUI, W., PEINADO, M., WANG, H. J., AND LOCASTO, M. E. Shieldgen: Automatic Data Patch Generation for Unknown Vulnerabilities with Informed Probing. In *Proceedings of the 28th IEEE Symposium on Security and Privacy* (2007), IEEE, pp. 252–266.

[7] LEE, B., LU, L., WANG, T., KIM, T., AND LEE, W. From zygote to morula: Fortifying weakened aslr on android. In *Proceedings of the 2014 IEEE Symposium on Security and Privacy* (2014), SP '14.

[8] PERKINS, J. H., KIM, S., LARSEN, S., AMARASINGHE, S., BACHRACH, J., CARBIN, M., PACHECO, C., SHERWOOD, F., SIDIROGLOU, S., SULLIVAN, G., WONG, W.-F., ZIBIN, Y., ERNST, M. D., AND RINARD, M. Automatically Patching Errors in Deployed Software. In *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles* (October 2009).

[9] SZEKERES, L., PAYER, M., WEI, T., AND SONG, D. Sok: Eternal war in memory. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy* (Washington, DC, USA, 2013), SP '13, IEEE Computer Society, pp. 48–62.

[10] TIAN, K., YAO, D., RYDER, B. G., AND TAN, G. Analysis of code heterogeneity for high-precision classification of repackaged malware. In *Security and Privacy Workshops (SPW), 2016 IEEE* (2016), IEEE, pp. 262–271.

[11] WANG, Z., WU, C., LI, J., LAI, Y., ZHANG, X., HSU, W.-C., AND CHENG, Y. Reranz: A light-weight virtual machine to mitigate memory disclosure attacks. In *Proceedings of the 13th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments* (2017), ACM, pp. 143–156.

[12] WARTELL, R., MOHAN, V., HAMLEN, K. W., AND LIN, Z. Binary stirring: Self-randomizing instruction addresses of legacy x86 binary code. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security* (2012), CCS '12.

[13] WIKIPEDIA. Pwn2Own. `http://en.wikipedia.org/wiki/Pwn2Own`.

[14] WILLIAMS-KING, D., GOBIESKI, G., WILLIAMS-KING, K., BLAKE, J. P., YUAN, X., COLP, P., ZHENG, M., KEMERLIS, V. P., YANG, J., AND AIELLO, W. Shuffler: Fast and deployable continuous code re-randomization. In *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation* (2016), USENIX Association, pp. 367–382.

[15] XU, J., NING, P., KIL, C., ZHAI, Y., AND BOOKHOLT, C. Automatic Diagnosis and Response to Memory Corruption Vulnerabilities. In *Proceedings of the 12th ACM conference on Computer and communications security* (November 2005), CCS '05.

[16] XU, K., YAO, D. D., RYDER, B. G., AND TIAN, K. Probabilistic program modeling for high-precision anomaly classification. In *Computer Security Foundations Symposium (CSF), 2015 IEEE 28th* (2015), IEEE, pp. 497–511.