Madani: Jurnal Ilmiah Multidisiplin
Volume 2, Nomor 8, Agustus 2024, Halaman 87-94

E-ISSN: 2986-6340
DOI: https://doi.org/10.5281/zenodo.13208360

Integrasi Address Space Layout Randomization (ASLR) Pada Kernel Linux X86 (32bit) Untuk Mencegah Serangan Exploit Buffer Overflow

Rakhmadi Rahman[1], Ayu Anugra[2], Widyah Putri Auliya Amin[3]
[123]Program Studi Sistem Informasi, Institut Teknologi Bacharuddin Jusuf Habibie
Email: ayu.paree2017@gmail.com

Abstrak

Buffer Overlow terdiri dari dua kata, yaitu buffer dan overflow. Buffer merujuk pada area penyimpanan data, sementara overflow berarti meluap. Secara sederhana, buffer overlow adalah situasi dimana area penyimpanan data menerima lebih banyak data daripada kapasitasnya. Bufer Overflow juga dapat diartikan sebagai metode yang sering digunakan leh hacker untuk mengeksploitasi sistem aplikasi komputer. Buffer Overflow dapat merusak program yang terkena serangan. Hal ini bisa menyebabkan program tidak berfungsi dengan benar, mengalami crash, atau bahkan tidak dapat dijalankan sama sekali. Kerusakan yang disebabkan oleh buffer overflow dapat mengganggu sistem operasi dan menyebabkan kehilangan data yang tidak dapat dipulihkan. Oleh karena itu, kita harus mencegah serangan exploit buffer overflow tersebut, salah satu cara yang dapat kita lakukan adalah dengan mengimplementasikan Address Space Layout Randomization (ASLR).
Kata Kunci: *Buffer Overflow, Address Space Layout Randomization, sistem aplikasi komputer, sistem operasi*

Abstract

Buffer Overflow consist of two words, namely buffer and overflow. Buffer refers to area data storage, while overflow means overflowing. In simple terms, buffer overflow is a situation where the data storage are receives more data then its capacity. Buffer Overflow can also be interpreted as a frequently used method by hackers to exploit computer application systems. Buffer Overflow can damage the affected program. This is possible causing the program to not function properly, crash, or not at all can be run at all. Damage caused by buffer overflow can disrupts the operating system and causes irrecoverable data loss. Therefore, we must prevent buffer overflow exploit attacks, one of them the way we can do this i by implementing Addresss Space Layout Randomization (ASLR).
Keywords: *Buffer Overflow; Address Space Layout Randomization, computer aplication systems, operating system*

Article Info
Received date: 25 July 2024 Revised date: 31 July 2024 Accepted date: 1 August 2024

PENDAHULUAN

Buffer overflow merupakan salah satu jenis kerentanan yang paling dikenal dalam dunia keamanan komputer. Istilah ini terdiri dari dua terjadi ketika area penyimpanan data menerima lebih banyak data daripada kapasitasnya, mengakibatkan data tersebut meluap ke area memori yang berdekatan. Kondisi ini dapat menyebabkan berbagai masalah pada program yang terkena serangan, mulai dari malfungsi, crash, hingga ketidakmampuan program untuk berjalan sama sekali. Kerentanan buffer overflow tidak hanya merusak program, tetapi juga memberikan celah bagi para hacker untuk mengeksploitasi sistem aplikasi komputer. Melalui metode ini, seorang hacker dapat menyuntikkan kode berbahaya ke dalam aplikasi dan mendapatkan kontrol atas sistem tersebut.

Dampak dari serangan buffer overflow sangatlah serius, karena dapat mengganggu kinerja menyebabkan crash, sistem dan operasi, bahkan mengakibatkan kehilangan data yang tidak dapat dipulihkan. Seiring dengan berkembangnya teknologi dan meningkatnya kompleksitas aplikasi, kebutuhan akan keamanan sistem komputer menjadi semakin mendesak. Salah satu pendekatan yang efektif untuk mencegah exploit buffer overflow adalah implementasi Address Space Layout Randomization (ASLR). ASLR adalah teknik keamanan yang mengacak tata letak ruang alamat memori, membuatnya lebih sulit bagi hacker untuk memprediksi lokasi kode berbahaya yang dapat dieksekusi. Dengan mengimplementasikan ASLR, kita dapat meningkatkan keamanan sistem dan mengurangi risiko serangan buffer overflow.

METODE PENELITIAN

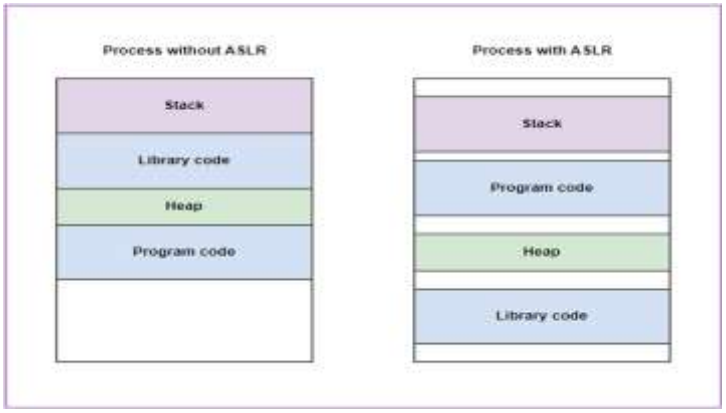
Penelitian dimulai dengan studi literatur untuk memahami konsep dasar dan perkembangan terkini ASLR serta cara kerjanya pada berbagai sistem operasi. Selanjutnya, dilakukan analisis terhadap struktur dan arsitektur kernel Linux untuk mengidentifikasi titik-titik kritis di mana ASLR dapat diintegrasikan secara efektif. Tahap berikutnya melibatkan pengembangan dan kata, yaitu "buffer" yang merujuk pada area penyimpanan data, dan "overflow" yang berarti meluap. Secara sederhana, buffer overflow modifikasi kode sumber kernel untuk menambahkan fitur ASLR, disertai dengan dokumentasi rinci setiap perubahan yang dilakukan. Setelah itu, dilakukan pengujian awal di lingkungan simulasi untuk memastikan bahwa modifikasi berfungsi dengan baik tanpa menyebabkan gangguan atau kerusakan sistem. Uji coba lanjutan dilakukan pada berbagai versi kernel Linux untuk menilai kompatibilitas dan stabilitas ASLR yang telah diintegrasikan. Data hasil pengujian kemudian dianalisis untuk mengevaluasi efektivitas dan efisiensi dari integrasi ASLR ini. Metode penelitian juga mencakup pengukuran kinerja sistem sebelum dan sesudah penerapan ASLR untuk memahami dampaknya terhadap performa keseluruhan. Selain itu, dilakukan memastikan tinjauan bahwa keamanan untuk ASLR benar-benar meningkatkan perlindungan terhadap eksploitasi keamanan. Metode yang dijelaskan tersebut merupakan gabungan antara pendekatan kualitatif dan kuantitatif.

HASIL DAN PEMBAHASAN

Address Space Layout Randomizition (ASLR)

Gambar 1 Enable ASLR

Address Space Layout Randomization (ASLR) merupakan teknik keamanan komputer yang terlibat dalam mencegah eksploitasi kerentanan kerusakan memori. Untuk mencegah penyerang mengalihkan eksekusi kode secara andal ke, misalnya, fungsi tertentu yang dieksploitasi dalam memori, ASLR secara acak mengatur posisi ruang alamat dari area data utama suatu proses , termasuk basis eksekusi dan posisi dari tumpukan , tumpukan dan perpustakaan. Hal ini dilakukan dengan menempatkan basis, perpustakaan, heap, dan tumpukan pada posisi acak di ruang alamat suatu proses.

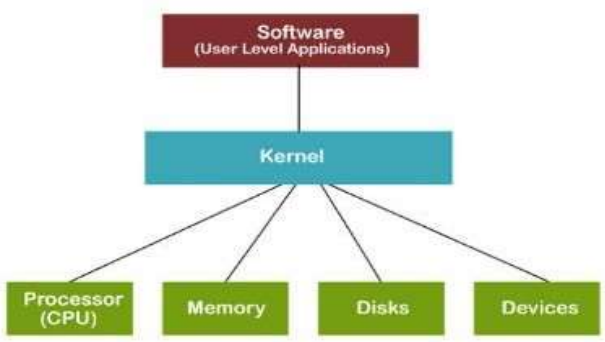


Gambar 2. Proses tanpa ASLR dan Proses dengan ASLR

Hal ini membuat tata letak memori tidak dapat diprediksi oleh program penyerang karena mereka tidak mengetahui di mana letak instruksi selanjutnya. Bagaimana Address Space Layout Randomization (ASLR) bekerja? ASLR diimplementasikan dalam beberapa bentuk di sebagian besar sistem operasi (OS), termasuk iOS, Android, Windows, macOS, dan Linux. Ini mencakup elemen seperti, stack; library code; Heap; Shared Memory Between Difference Processes; Command Line Arguments. ASLR bekerja bersama dengan manajer memori virtual untuk mengacak lokasi elemen program yang disebutkan di atas dalam memori. Pengacakan ini terjadi setiap kali program dijalankan, menyebabkan alamat bervariasi secara konsisten. Hal ini mencegah penyerang memasukkan kode berbahaya, karena mereka tidak dapat lagi menebak alamat program melalui trial and error. Meskipun ASLR adalah cara yang baik untuk membantu membentengi program terhadap eksploitasi buffer yang umum, ada beberapa cara yang terbukti merugikan seperti: ASLR tidak memunculkan peringatan ketika upaya untuk melewatinya terjadi; Setelah serangan berhasil, ASLR tidak memberikan informasi apa pun tentang serangan tersebut, seperti akses memori atau panggilan ke tumpukan; Beberapa demonstrasi yang berhasil menunjukkan kerentanan ASLR dengan memprediksi hasil pengacakan. Namun, dukungan perangkat keras yang lebih kuat dapat membantu mengurangi kerentanan ini.

Kernel

Kernel merupakan landasan dasar untuk sebuah sistem operasi atau OS (Operating System). Sistem ini menjadi lapisan penghubung antara perangkat keras dari komputer dengan OS yang bekerja di dalamnya. Tugas utama dari kernel adalah untuk mendorong kerja hardware untuk menjalankan aplikasi dan OS (Operating System), lalu memuatnya ke dalam memori utama. Sistem ini mendorong kerja hardware dengan cara mengelola sistem dalam memory management, process dan dan task management. Selain itu kernel juga bekerja dalam pengelolaan disk management.



Gambar 3. Kernel pada Sistem Operasi

Ada 4 kategori kernel:

a) Kernel menolitik. Kernel yang menyediakan abstraksi perangkat keras yang kaya dan tangguh.
b) keras sederhana,dan menggunakan aplikasi-aplikasi yang disebut sebagaai server untuk menyediakan fungsi fungsi Mikrokernel. Kernel yang hanya menyediakan sekumpulan abstraksi kecil perangkat lainnya.
c) Hybrid (modifikasi dari mikrokernel). Kernel yang mirip dengan microkerne, tetapi iya juga memasukkan beberapa kode tambahan di kernel agar menjadi lebih cepat.
d) Eksokernel. Kernel yang tidak menyediakan sama sekali abstraksi perangakat keras,tetapi iya menyediakan sekumpulan pustaka yang menyediakan fungsi-fungsi akses ke perangkat keras secara langsung atau hampir-hampir langsung.

Fungsi utama kernel:

a) Manajemen Memori: Mengatur alokasi dan penggunaan memori untuk aplikasi dan proses yang berjalan.
b) Manajemen Proses: Mengatur dan mengontrol proses yang berjalan di komputer, termasuk penjadwalan, sinkronisasi, dan komunikasi antar proses.
c) Manajemen Perangkat Keras: Mengatur dan mengontrol akses aplikasi ke perangkat keras, seperti CPU, disk, dan perangkat I/O lainnya.
d) Keamanan Sistem: Memberikan kontrol akses dan perlindungan terhadap sumber daya sistem dari akses yang tidak sah.
e) Layanan Sistem: Menyediakan layanan dasar untuk aplikasi, seperti sistem panggilan,

interupsi, dan penanganan sinyal.

Bagaimana kernel bekerja:

a) Bootloader: Saat komputer dinyalakan, bootloader memuat kernel ke dalam memori.
b) Nisialisasi: Kernel menginisialisasi sistem, termasuk mengatur memori, perangkat keras, dan tabel routing.
c) Menjalankan Proses: Kernel menjalankan proses sesuai dengan algoritma penjadwalan ditentukan.
d) Menangani Interupsi: Kernel menangani interupsi dari perangkat keras dan perangkat lunak, seperti penekanan tombol atau sinyal dari aplikasi.
e) Menyediakan Layanan Sistem: Kernel menyediakan layanan sistem untuk aplikasi, seperti sistem panggilan dan penanganan sinyal.

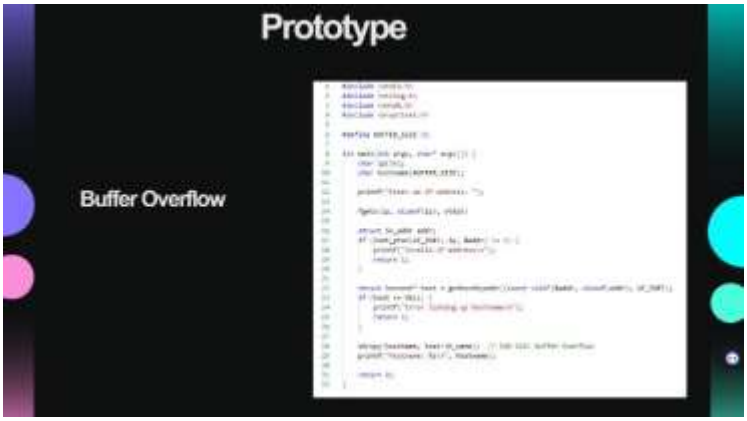
Linux

Linux adalah keluarga sistem operasi mirip Unix bebas dan sumber terbuka yang didasarkan pada kernel Linux, yaitu sebuah kernel sistem operasi yang pertama kali dikembangkan oleh Linus Torvalds pada 1991. Linux dirilis di bawah Lisensi Publik Umum GNU versi 2. Linux aslinya dikembangkan untuk komputer pribadi berarsitektur Intel x86, tetapi seiring waktu Linux telah diportasi ke berbagai arsitektur, lebih banyak daripada sistem operasi lainnya. Secara teknis, Linux dapat merujuk pada kernel-nya itu sendiri. Kernel Linux bermacam yang dilengkapi dengan perangkat lunak lainnya membentuk sebuah distribusi Linux. Beberapa orang, terutama dari anggota Yayasan Perangkat Lunak Bebas (FSF) seperti Richard Stallman, merujuk Linux sebagai GNU/Linux karena banyak alat-alat yang digunakan untuk menunjang utilitasnya berasal dari Proyek GNU besutan Stallman. Ini memunculkan kontroversi terkait nama tersebut. Sejak pertengahan 1990-an hingga 2000-an, Linux menjadi sistem operasi yang mendominasi di pasar peladen, komputasi awan, dan superkomputer (dan sejak 2017, Linux adalah sistem operasi yang satu-satunya digunakan dalam daftar superkomputer TOP500). Linux juga mendominasi di pasar ponsel melalui sistem operasi Android buatan Google, yang digunakan sekitar 72.7 persen secara global per 2021. Dalam beberapa tahun terakhir, Linux semakin banyak digunakan di komputer pribadi meskipun Windows tetap mendominasi. Linux digunakan oleh berbagai macam pengguna, termasuk pengguna pribadi, pemerintah, hingga organisasi dan perusahaan. Perbedaan utama antara Linux dan sistem operasi populer lainnya terletak pada kernel Linux dan komponen-komponennya yang bebas dan terbuka. Linux bukan satu-satunya sistem operasi dalam kategori tersebut, walaupun demikian Linux adalah contoh terbaik dan terbanyak digunakan.

Beberapa lisensi perangkat lunak bebas dan sumber terbuka berdasarkan prinsip-prinsip copyleft, sebuah konsep yang menganut prinsip: karya yang dihasilkan dari bagian copyleft harus juga merupakan copyleft. Lisensi perangkat lunak bebas yang paling umum, GNU GPL, adalah sebuah bentuk copyleft, dan digunakan oleh kernel Linux dan komponen-komponen dari proyek GNU. Kebanyakan sistem Linux patuh pada standar POSIX, SUS, ISO, dan ANSI. EulerOS dan Inspur K-UX, masing-masing dibuat oleh Huawei dan Inspur, adalah distribusi Linux yang disertifikasi POSIX dan SUS.

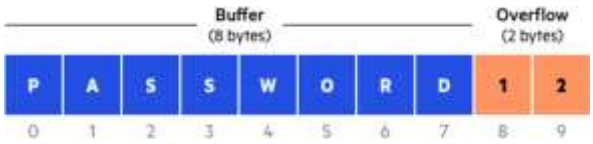
Proyek-proyek perangkat lunak bebas, walaupun dikembangkan dalam bentuk kolaborasi, sering dirilis secara terpisah. Akan tetapi, dikarenakan lisensi-lisensi perangkat lunak bebas secara eksplisit mengizinkan distribusi ulang, terdapat proyek-proyek yang bertujuan untuk mengumpulkan perangkat lunak-perangkat lunak menjadikannya tersedia tersebut dan dalam waktu bersamaan dalam suatu bentuk yang dinamakan distribusi Linux. Sebuah distribusi Linux, yang umum disebut dengan "distro", adalah sebuah proyek yang bertujuan untuk mengatur sebuah kumpulan perangkat lunak berbasis Linux dan memfasilitasi instalasi dari sebuah sistem operasi Linux. Distribusi distribusi Linux ditangani oleh individu, tim, organisasi sukarelawan dan entitas komersial. Distribusi Linux memiliki perangkat lunak sistem dan aplikasi dalam bentuk paket-paket dan perangkat lunak yang spesifik dirancang untuk instalasi dan konfigurasi sistem. Perangkat lunak tersebut juga bertanggung jawab dalam pemutakhiran paket. Sebuah Distribusi Linux bertanggung jawab atas konfigurasi bawaan, sistem keamanan dan integrasi secara umum dari paket-paket perangkat lunak sistem Linux.

Buffer Ovaerflow



Gambar 4 Buffer Overflow

Buffer overflow terdiri dari dua kata, yaitu buffer dan overflow. Dalam hal ini, buffer diartikan sebagai tempat penyimpanan data. Sementara overflow artinyakelebihan aliran. Jadi, simpelnya buffer overflow adalah kondisi di mana sebuahtempat penyimpanan data mengalami kelebihan muatan. Buffer overflow jugadapat dikatakan sebagai sebuah metode yang biasa digunakan oleh hacker untuk melakukan eksploitasi pada sebuah sistem aplikasi komputer. Tindakan ini juga biasa disebut dengan serangan input validasi, dimana hacker akan melakukan inputdata yang melebihi kapasitas penyimpanan suatu aplikasi, sehingga membuatsistem mengalami crash atau buffer overflow.



Gambar 5 Buffer Overlfow

Penyerang biasanya menggunakan kombinasi data input yang dibuat khusus dankode berbahaya untuk mengeksploitasi kerentanan dalam perangkat lunak sistemyang ditargetkan. Kode jahat memanipulasi buffer, sehingga meluap danmemungkinkan penyerang mengeksekusi kode ini.Untuk melakukan serangan penyerang buffer overflow, pertama-tamamengidentifikasi sistem atau aplikasi perangkat lunak yang rentan dan membuatmuatan data yang dirancang untuk mengeksploitasi kerentanan tersebut. Jaringanatau vektor serangan berbasis web, seperti situs web atau email jahat, mengirimkanmuatan. Sistem target menerima muatan dan memproses aplikasi perangkat lunak,yang berupaya menyimpan data yang masuk dalam buffer. Jika buffer tidak cukup besar untuk menampung data, buffer akan meluap kodedieksekusi mestinya.Penyerang dan memungkinkan sebagaimana kemudian dapat menguasai sistem dan berpotensi mencuri data sensitif,mengganggu operasi, atau mendapatkan akses ke sistem tambahan di jaringan. Penting untuk memperbarui aplikasi perangkat lunak secara teratur danmenerapkan langkah-langkah keamanan seperti firewall dan sistem deteksi intrusiuntuk mencegah serangan buffer overflow.

Sebelum mempelajari jenis-jenis serangan buffer overflow, ada beberapa insiden populer serangan buffer dari sejarah.

1. Salah satu worm komputer pertama yang menerima sejumlah besar perhatianmedia arus utama adalahworm Morris 2 November 1988, sekarang dikenalsebagai Internet. worm Serangan worm Morris mengeksploitasi beberapakerentanan, termasuk UNIX (menggunakan (melalui buffer backdoor), sendmail finger overflow), dan rsh/rexec. Selain itu, ia dapat menebak katasandi yang lemah.
2. Pada November 2014, perusahaanSony Pictures Entertainment mengalami kerusakan besar pada sistem komputernya yang disebabkan oleh serangan buffer overflow. Penyerang mencuri informasi sensitif, termasuk film yang belum dirilis dan data pribadi karyawan dan selebritas.
3. Pada Juni 2011, bank Citigroup mengalami serangan buffer overflow yangmemberi peretas akses ke informasi pribadi lebih dari 200.000 pelanggan,termasuk nama, alamat, dan nomor rekening mereka. Para penyerangmenggunakan informasi ini untuk mencuri lebih dari $2,7 juta dari bank.

4. Pengembang Libgcryptmengeluarkan pembaruan tambalan keamanan padaJanuari 2021. Setelah mereka menemukan kerentanan buffer overflow berbasis heap.

Hacker dapat menggunakan cara yang berbeda untuk menyerang sebuah sistemdan membuatnya mengalami buffer overflow. Berikut ini adalah beberapa jenisserangan buffer overflow yang paling terkenal.1.Serangan Stack Overflow ini merupakan jenis.

1. Serangan Stack Overflow
   Ini merupakan jenis serangan buffer overflow yang paling banyak digunakanoleh hacker. Stack adalah struktur data LIFO (Last In First Out) yangmendukung dua operasi, yaitu PUSH dan POP. Operasi PUSH digunakanuntuk nilai data, sedangkan POP digunakan untuk mengekstrak nilai data.Apabila penyimpanan data pada stack rusak, maka data yang masuk akan disimpan pada alamat penyimpanan yang berdekatan. Hal ini dapatmemengaruhi data yang disimpan sebelumnya. Jenis serangan ini biasanyaterjadi karena adanya kerentanan pada sistem dengan bahasa pemrograman C atau C++.
2. Heap Overflow Attack
   Hacker biasanya akan menargetkan data yang terdapat pada open memory pool, atau dikenal sebagai heap. Heap overflow dapat terjadi ketika sebuahmemori ditetapkan sebagai heap, kemudian data disimpan dalam memoritersebut tanpa dilakukan pemeriksaan. Hal tersebut dapat mengakibatkan beberapa struktur penting dalam heap, seperti heap reader, atau data berbasisheap lainnya seperti dynamic object pointer melakukan overwrite pada tabelfungsi virtual.
3. Integer Overflow Attack
   Integer overflow merupakan jenis kesalahan aritmatika yang dapat membuat operasi integer tidak berada pada penyimpanan yang dialokasikan. Padakebanyakan bahasa pemrograman, nilai integer biasanya memiliki beberapa jumlah bit tertentu dalam penyimpanan. Umumnya, kondisi integer overflowhanya akan menyebabkan kesalahan pada bahasa pemrograman dan tidakakan mengakibatkan kerentanan yang signifikan. Namun, pada beberapakasus, integer overflow dapat menyebabkan risiko yang parah sepertimemanipulasi data keuangan.
4. Unicode Overflow
   Unicode overflow dapat menyebabkan buffer overflow dengan caramemasukkan karakter unicode dalam input ASCII (American Standard Codefor Information Interchange) yang diharapkan. Kasus penyerangan bufferoverflow ini sangat jarang sekali terjadi di Indonesia. Meskipun begitu,Anda tetap harus mewaspadainya karena cyber attack bisa datang kapan dandari mana saja.

Serangan buffer overflow dapat menyebabkan kerusakan signifikan pada organisasi dan meningkatkan risiko kerentanan keamanan. Berikut adalah beberapa konsekuensi dari mengalami serangan buffer overflow.

1. Ketidakstabilan sistem
2. Hilangnya kontrol akses
3. Kerugian data atau fungsial
4. Jalankan kode arbitrer
5. Sistem keamanan yang disusupi
6. Kerusakan reputasi

Gambar 6 Implementing Address Space Layout Randomization on Kernel

a. Persiapan Lingkungan Pengembangan: Instalasi dependensi yang diperlukan dan unduh serta ekstrak kode sumber kernel.
b. Modifikasi Kode Kernel: Melakukan perubahan pada fs/binfmt_elf.c untuk randomisasi alamat stack dan mm/mmap.c untuk randomisasi base address mmap.
c. Konfigurasi Kernel: Menggunakan make menuconfig untuk mengaktifkan pengaturan ASLR di kernel. Kompilasi Kernel: Kompilasi kernel dan instalasi modul serta kernel baru.
d. Konfigurasi Sistem: Mengaktifkan ASLR pada sistem melalui /proc/sys/kernel/randomize_va_space.
e. Pengujian dan Debugging: Boot ke kernel baru, verifikasi implementasi ASLR, dan debugging jika ada masalah.
f. Dokumentasi Perubahan: Mencatat perubahan yang dilakukan, membuat panduan penggunaan, dan mencatat hasil pengujian.

Ketika ASLR (Address Space Layout Randomization) telah diimplementasikan pada kernel Linux, beberapa perubahan penting terjadi pada cara sistem mengalokasikan dan mengelola memori untuk proses.

SIMPULAN

Integrasi ASLR pada kernel Linux terbukti efektif dalam meningkatkan keamanan sistem terhadap serangan buffer overflow. Dengan mengacak lokasi memori yang digunakan oleh program, ASLR secara signifikan mengurangi kemungkinan penyerang dapat memprediksi lokasi spesifik untuk menyuntikkan kode berbahaya. Pengujian yang dilakukan menunjukkan bahwa setelah implementasi ASLR, kerentanan terhadap exploit buffer overflow berkurang drastis. Selain itu, hasil pengukuran kinerja menunjukkan bahwa penerapan ASLR hanya memberikan dampak minimal terhadap performa sistem, sehingga tidak mengganggu operasi normal. Analisis keamanan menunjukkan bahwa ASLR bekerja dengan baik bersama mekanisme keamanan lainnya di kernel Linux, seperti stack canaries dan Data Execution Prevention (DEP), untuk menyediakan lapisan perlindungan tambahan. Meskipun demikian, penting untuk diingat bahwa ASLR bukanlah solusi tunggal dan harus digunakan bersama dengan praktik keamanan lainnya untuk mencapai perlindungan maksimal. Kesimpulannya, integrasi ASLR pada kernel Linux adalah langkah strategis yang efektif untuk memperkuat keamanan sistem terhadap serangan buffer overflow, sambil mempertahankan kinerja dan stabilitas.

REFERENSI

Aga, M.T.; Austin, T. Smokestack: Thwarting DOP Attacks with Runtime Stack LayoutRandomization. In Proceedings of the 2019 IEEE/ACM International Symposium on Code Generation and Optimization (CGO), Washington, DC, USA, 16–20 February 2019; pp. 26–36.

[Google Scholar] [CrossRef]
Jelinek, J. Object Size Checking to Prevent (Some) Buffer Overflows (GCC FORTIFY). 2004. Available online: http://gcc.gnu.org/ml/gcc-patches/2004 09/msg02055.html
Pax Team. PaX Address Space Layout Randomization (ASLR). 2003. Available online: http://pax.grsecurity.net/docs/aslr.txt (accessed on 17 July 2019).
Edge, J. Kernel Address Space Layout Randomization. 2013. Available online: https://lwn.net/Articles/569635
Lefevre, V. Silent Stack-Heap Collision under GNU/Linux. 2014. Available online: https://gcc.gnu.org/ml/gcc-help/2014 07/msg00076.html (accessed on 17 July 2019).