Tìm hiểu về Linux Kernel và những chức năng chính của nó

Với hơn 13 triệu dòng lệnh, Linux kernel là 1 trong những dự án mã nguồn mở rộng lớn nhất trên thế giới, nhưng chính xác chúng là gì và chúng làm gì trong hệ thống?

Kernel là gì?

Khái niệm kernel ở đây nói đến những phần mềm, ứng dụng ở mức thấp (*low-level*) trong hệ thống, có khả năng thay đổi linh hoạt để phù hợp với phần cứng. Chúng tương tác với tất cả ứng dụng và hoạt động trong chế độ user mode, cho phép các quá trình khác – hay còn gọi là server, nhận thông tin từ các thành phần khác qua inter-process communication (IPC).

Các loại kernel khác nhau

Về bản chất, có nhiều cách để xây dựng cấu trúc và biên dịch 1 bộ kernel nhất định từ đầu. Nhìn chung, với hầu hết các kernel hiện nay, chúng ta có thể chia ra làm 3 loại: *monolithic*, *microkernel*, và *hybrid*. Linux sử dụng kernel monolithic trong khi OS X (XNU) và Windows 7 sử dụng kernel hybrid.

Microkernel

Microkernel có đầy đủ các tính năng cần thiết để quản lý bộ vi xử lý, bộ nhớ và IPC. Có rất nhiều thứ khác trong máy tính có thể được nhìn thấy, tiếp xúc và quản lý trong chế độ người dùng. Microkernel có tính linh hoạt khá cao, vì vậy bạn không phải lo lắng khi thay đổi 1 thiết bị nào đó, ví dụ như card màn

hình, ổ cứng lưu trữ... hoặc thậm chí là cả hệ điều hành. Microkernel với những thông số liên quan footprint rất nhỏ, tương tự với bộ nhớ và dung lượng lưu trữ, chúng còn có tính bảo mật khá cao vì chỉ định rõ ràng những tiến trình nào hoạt động trong chế độ user mode, mà không được cấp quyền như trong chế độ giám sát - supervisor mode.

Ưu điểm:
- Tính linh hoạt cao
- Bảo mật
- Sử dụng ít footprint cài đặt và lưu trữ

Nhược điểm:
- Phần cứng đôi khi "khó hiểu" hơn thông qua hệ thống driver
- Phần cứng hoạt động dưới mức hiệu suất thông thường vì các trình điều khiển ở trong chế độ user mode
- Các tiến trình phải chờ đợi để được nhận thông tin
- Các tiến trình không thể truy cập tới những ứng dụng khác mà không phải chờ đợi

Monolithic Kernel

Với Monolithic thì khác, chúng có chức năng bao quát rộng hơn so với microkernel, không chỉ tham gia quản lý bộ vi xử lý, bộ nhớ, IRC, chúng còn can thiệp vào trình điều khiển driver, tính năng điều phối file hệ thống, các giao tiếp qua lại giữa server... Monolithic tốt hơn khi truy cập tới phần cứng và đa tác vụ, bởi vì nếu 1 chương trình muốn thu thập thông tin từ bộ nhớ và các tiến trình khác, chúng cần có quyền truy cập trực tiếp và không phải chờ đợi các tác vụ khác kết thúc. Nhưng đồng thời, chúng cũng là nguyên nhân gây ra sự bất ổn vì nhiều chương trình chạy trong chế độ supervisor mode hơn, chỉ cần 1 sự cố nhỏ cũng khiến cho cả hệ thống mất ổn định.

Ưu điểm:
- Truy cập trực tiếp đến các phần cứng
- Dễ dàng xử lý các tín hiệu và liên lạc giữa nhiều thành phần với nhau
- Nếu được hỗ trợ đầy đủ, hệ thống phần cứng sẽ không cần cài đặt thêm

driver cũng như phần mềm khác
- Quá trình xử lý và tương tác nhanh hơn vì không cần phải chờ đợi

Nhược điểm:
- Tiêu tốn nhiều footprint cài đặt và lưu trữ
- Tính bảo mật kém hơn vì tất cả đều hoạt động trong chế độ giám sát - supervisor mode

Hybrid Kernel

Khác với 2 loại kernel trên, Hybrid có khả năng chọn lựa và quyết định những ứng dụng nào được phép chạy trong chế độ user hoặc supervisor. Thông thường, những thứ như driver và file hệ thống I/O sẽ hoạt động trong chế độ user mode trong khi IPC và các gói tín hiệu từ server được giữ lại trong chế độ supervisor. Tính năng này thực sự rất có ích vì chúng đảm bảo tính hiệu quả của hệ thống, phân phối và điều chỉnh công việc phù hợp, dễ quản lý.

Ưu điểm:
- Các nhà phát triển có thể chọn và phân loại những ứng dụng nào sẽ chạy trong chế độ thích hợp
- Sử dụng ít footprint hơn so với monolithic kernel
- Có tính linh hoạt và cơ động cao nhất

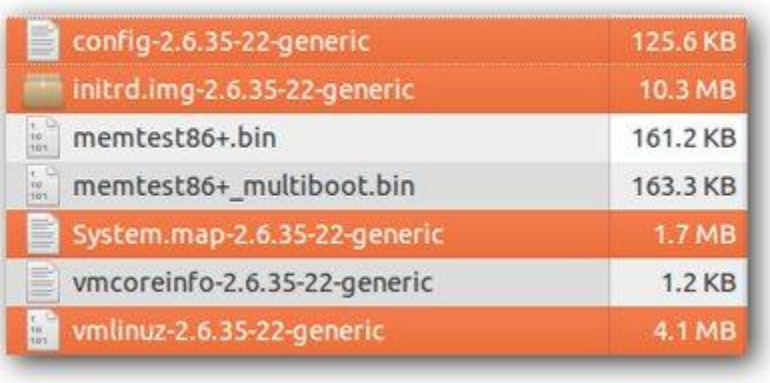
Nhược điểm:
- Có thể bị bỏ lại trong quá trình gây treo hệ thống tương tự như với microkernel
- Các trình điều khiển thiết bị phải được quản lý bởi người dùng

Vậy những file Linux Kernel này ở đâu?

Các file kernel này, trong Ubuntu chúng được lưu trữ tại thư mục /boot và đặt tên theo vmlinuz-version. Khi bộ nhớ ảo bắt đầu được phát triển để thực hiện các tác vụ đa luồng, tiền tố vm sẽ được đặt vào đầu các file kernel để phân biệt khả năng hỗ trợ công nghệ ảo hóa. Kể từ đó, Linux kernel được gọi là vmlinux, nhưng hệ thống kernel này đã phát triển với tốc độ quá nhanh, lớn

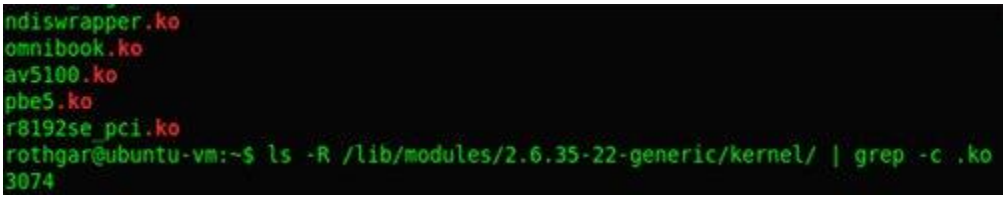
hơn so với dung lượng bộ nhớ boot chuẩn của hệ điều hành, vì vậy những file kernel này đã được nén theo chuẩn zlib – và ký tự z được thêm vào là do như vậy. Ngoài ra còn 1 số định dạng nén thường gặp khác là LZMA hoặc BZIP2, nhưng chúng vẫn được gọi chung là zImage.

Các phiên bản được sắp xếp thứ tự theo định dạng A.B.C.D, trong đó A.B thường là 2.6, C đại diện cho phiên bản, và D là ký hiệu các bản vá lỗi hoặc patch:



Trong thư mục /boot còn có rất nhiều file quan trọng khác như, initrd.img-version, system.map-version, và config-version. File initrd được dùng như 1 ổ đĩa RAM để giải nén và kích hoạt các file kernel thực sự, còn file system.map được dùng để quản lý bộ nhớ trước khi kernel được tải đầy đủ, và file config làm nhiệm vụ thông báo cho kernel biết những lựa chọn hoặc module nào sẽ được nạp vào quá trình hệ thống khởi động.

Cấu trúc file Linux Kernel



Thực tế, Windows đã có tất cả các trình điều khiển sẵn có và người sử dụng chỉ việc kích hoạt các trình điều khiển tương ứng để sử dụng. Và đó cũng chính là nhiệm vụ các module kernel Linux đảm nhiệm, hay còn được gọi là

loadable kernel module (LKM), rất cần thiết để giữ các chức năng đi kèm với toàn bộ hệ thống phần cứng hoạt động mà không ảnh hưởng đến bộ nhớ. 1 module thông thường sẽ gán chức năng cơ bản tới các kernel như điều khiển driver, file hệ thống... LKM có phần đuôi mở rộng là .ko và được lưu trữ trong thư mục /lib/modules, người sử dụng có thể thiết lập thuộc tính tự khởi động, cho phép tải hoặc không trong khi hệ điều hành khởi động, bằng cách dùng lệnh menuconfig, can thiệp vào file /boot/config, hoặc bằng cách sử dụng lệnh modprobe.

Các module của các hãng thứ 3 thường có sẵn trên 1 số distributor, ví dụ như Ubuntu, hoặc không được tích hợp sẵn ở chế độ mặc định. Các nhà phát triển phần mềm (ví dụ nVidia, ATI hoặc các hãng khác), không cung cấp mã nguồn nhưng họ đã tự xây dựng và biên dịch các module cần thiết, sau đó cung cấp cho người sử dụng file .ko. Và tất nhiên, có nhiều module hoàn toàn miễn phí, trong khi một số khác thì không.

Hai lỗ hổng nghiêm trọng trong Linux Kernel

Các nhà nghiên cứu bảo mật đã tiết lộ về hai lỗ hổng nghiêm trọng trong Linux Kernel, cho phép kẻ tấn công có được đặc quyền root trên các hệ thống Linux.

Lỗ hổng đầu tiên được phát hiện bởi các nhà nghiên cứu từ hãng bảo mật Qualys và được theo dõi dưới tên gọi CVE-2018-14634. Lỗ hổng này nằm trong hàm **create_elf_tables**() của Linux Kernel và có thể được khai thác trên các hệ thống 64 bit của người dùng cục bộ, có quyền truy cập vào các file nhị phân SUID.

Theo các nhà nghiên cứu của Qualys, người đã đặt tên cho lỗ hổng này là Mutagen Astronomy. Lỗi này đã tồn tại trong Linux Kernel khoảng một thập kỷ. Bản sửa lỗi **b6a2fea39318** đã được giới thiệu vào ngày 19 tháng 7 năm 2007. Một bản sửa lỗi khác mang tên **da029c11e6b1** tiếp tục được công bố vào ngày 7 tháng 7 năm 2017.

Tuy nhiên, thực tế là không phải tất cả các bản phân phối Linux đều dùng bản sửa lỗi da029c11e6b1 cho kernel của mình và vẫn còn dễ bị tấn công. Các bản phân phối bị ảnh hưởng bởi lỗ hổng này bao gồm Red Hat Enterprise Linux (RHEL) 6, 7 và Red Hat Enterprise MRG 2, cũng như CentOS, dựa trên RHEL và Debian 8 Jessie (oldstable).

"Vấn đề này không ảnh hưởng đến các hệ thống 32-bit vì chúng không có đủ không gian địa chỉ để khai thác lỗ hổng này", Red Hat phát biểu trong một cuộc tư vấn bảo mật. "Các hệ thống có bộ nhớ ít hơn 32GB rất khó bị ảnh hưởng bởi vấn đề này do nhu cầu bộ nhớ trong quá trình khai thác".

Red Hat có kế hoạch khắc phục vấn đề này trong bản cập nhật kernel trong tương lai, nhưng cho đến lúc đó, có những giải pháp thủ công để bảo vệ các hệ thống chống lại việc bị khai thác. Giải pháp thay thế cần phải được áp dụng một lần nữa sau khi khởi động lại hệ thống.

Lỗ hổng thứ hai, có tên gọi là **CVE-2018-17182,** được tìm thấy bởi Jann Horn, một nhà nghiên cứu bảo mật trong dự án Project Zero của Google. Lỗ hổng này cũng có thể được khai thác để thực thi code tùy ý dưới dạng root và ảnh hưởng đến tất cả các phiên bản kernel kể từ 3.16.

Horn cho thấy lỗ hổng có thể được khai thác như thế nào trên các kernel không được cấu hình để tăng cường bảo mật, nhưng cảnh báo rằng với nỗ lực bổ sung, code có thể kích hoạt lỗi này ngay trong sandbox: Seccomp. Seccomp là sandbox của thành phần lưu trữ gVisor chính và policy seccomp của Docker.

Horn phát biểu trong một bài đăng trên blog: "Để khiến mọi thứ dễ dàng hơn, khai thác của tôi sử dụng các giao diện kernel khác nhau, và do đó không chỉ

hoạt động từ bên trong các sandbox như vậy. Đặc biệt, nó sử dụng **/dev/kmsg** để đọc các bản ghi dmesg và sử dụng một mảng eBPF để spam công cụ phân bổ trang của kernel (việc phân bổ một trang có thể thay đổi được) do người dùng kiểm soát". "Tuy nhiên, một kẻ tấn công sẵn sàng đầu tư thêm thời gian vào một cuộc khai thác sẽ tránh được việc sử dụng các giao diện như vậy".

Lỗi này được báo cáo cho các nhà bảo trì Linux kernel vào ngày 12 tháng 9 và một bản sửa lỗi được tạo ra hai ngày sau đó, cực kỳ nhanh so với thời gian sửa lỗi của các nhà cung cấp phần mềm khác.

Tuy nhiên, ông cũng chỉ ra một vấn đề nổi tiếng trong hệ sinh thái Linux, được nhấn mạnh bởi lỗ hổng Mutagen Astronomy: Khi một lỗ hổng được vá trong Linux kernel, điều đó không có nghĩa là hệ thống của người dùng đã được bảo vệ.

Trong thực tế, có thể mất một khoảng thời gian dài cho đến khi hệ thống của những người dùng cuối nhận được bản vá. Đó là bởi vì hầu hết người dùng sử dụng một bản phân phối Linux cụ thể và dựa vào việc nhận các bản sửa lỗi bảo mật thông qua nó. Các bản phân phối Linux này thường sử dụng các kernel ổn định, bao gồm cả các kernel cũ hơn, vì vậy chúng cần phải chờ cho đến khi các nhà bảo trì đưa ra các bản sửa lỗi.

Bất kỳ sự chậm trễ nào cũng có thể mở ra một cánh cửa cho những kẻ tấn công, và đôi khi, giống như trong trường hợp của Mutagen Astronomy, một số bản vá có thể không quan trọng đối với một bản phân phối ở thời điểm hiện tại, lại có thể mang đến ý nghĩa bảo mật nhiều năm sau đó.

Bản vá cho lỗ hổng CVE-2018-17182 của Horn đã được quay trở lại các kernel 4.18, 4.14, 4.9 và 4.4 vào ngày 19 tháng 9, 4 ngày sau khi phát hiện lỗ hổng, và trở thành bản vá chính thức. Tuy nhiên, khi Horn xuất bản bài đăng chi tiết của mình trên blog, vào ngày 26 tháng 9, Debian vẫn sử dụng kernel 4.9 chưa được cập nhật kể từ ngày 21 tháng 8, trong khi Ubuntu 16.04 chuyển sang một kernel chưa được cập nhật kể từ tháng 27 tháng 8.

Như vậy, các bạn đã có thể hình dung được sự quan trọng của kernel. Kernel của Linux khác hẳn so với Mac OS X và Windows bởi trình điều khiển driver cũng như cách thức quản lý và hỗ trợ khác. Trên đây là 1 số thông tin cơ bản và cần thiết giúp mọi người có thể hiểu được kernel là gì, chúng hoạt động như thế nào và tại sao chúng lại cần thiết như vậy.