**Báo cáo thực hành: Thao túng và che giấu hoạt động độc hại bằng Rootkit trên Linux**

**1. Mục tiêu bài thực hành**

Bài thực hành này nhằm mục đích giúp sinh viên:

* Hiểu rõ **cơ chế hoạt động của rootkit**, đặc biệt là cách chúng thao tác nhân (kernel) của hệ điều hành Linux để che giấu các hoạt động bất hợp pháp.
* Nắm bắt được cách một rootkit đơn giản có thể **ẩn tệp, tiến trình và kết nối mạng**.
* Nhận thức được **các rủi ro bảo mật nghiêm trọng** khi hệ thống bị nhiễm rootkit.
* Có cái nhìn sơ bộ về **cách phát hiện và ngăn chặn** rootkit trong thực tế.

**2. Yêu cầu kiến thức nền tảng**

Để hoàn thành bài thực hành một cách hiệu quả, sinh viên cần có các kiến thức cơ bản sau:

* **Hệ điều hành Linux**: Hiểu biết về cấu trúc thư mục, quản lý tệp, tiến trình và quyền người dùng.
* **Kernel module (Mô-đun nhân)**: Khái niệm, cách thêm/gỡ mô-đun trên Linux (ví dụ: insmod, rmmod, lsmod).
* **Ngôn ngữ lập trình C**: Khả năng đọc và hiểu mã C cơ bản, đặc biệt là các khái niệm liên quan đến lập trình kernel.
* **Lệnh và công cụ bảo mật**: Sử dụng thành thạo các lệnh như ls, cat, ps, netstat, dmesg để kiểm tra trạng thái hệ thống và nhật ký.
* **Quy trình phát hiện rootkit**: Khả năng phân tích dấu hiệu nhiễm mã độc và hiểu các bước xử lý ban đầu.

**3. Nội dung và Quy trình thực hiện**

**a. Khởi động môi trường Lab**

1. Mở Terminal và gõ lệnh sau để khởi động môi trường Labtainer:

Bash

labtainer -r pen\_attk\_mitre\_t1014

1. Khi được yêu cầu, nhập **mã sinh viên** của bạn làm thông tin email người thực hiện bài lab.
2. Sau khi khởi động thành công, một Terminal ảo sẽ xuất hiện, đại diện cho máy **victim** (máy nạn nhân).

**b. Thay đổi mã nguồn Rootkit**

1. Tìm đến đoạn mã nguồn của rootkit (thường là một file .c hoặc .h trong thư mục lab) và thực hiện thay đổi để rootkit có thể **ẩn đúng file được yêu cầu**.
   * *Gợi ý*: Sinh viên sẽ cần chỉnh sửa một hàm hoặc một biến trong mã nguồn rootkit để chỉ định tên tệp cụ thể mà rootkit sẽ che giấu. Điều này thường liên quan đến việc móc nối (hooking) các lời gọi hệ thống (system calls) như sys\_getdents64 (để ẩn tệp khỏi ls) hoặc sys\_kill (để ẩn tiến trình khỏi ps).

Notes: Thay đổi

A screenshot of a computer

AI-generated content may be incorrect.

**c. Biên dịch Rootkit**

1. Sau khi thay đổi mã nguồn, lưu lại file.
2. Trong Terminal của máy victim, điều hướng đến thư mục chứa mã nguồn rootkit và thực hiện biên dịch bằng lệnh:

Make

Notes:

Your uname -r reports the kernel version as **6.11.0-29-generic**. However, when you list /usr/src/, the only kernel headers present are for **4.18.0-15** and **4.18.0-15-generic**.

This is a **kernel version mismatch**. Your make command is trying to use the build environment for 6.11.0-29-generic, but those headers don't exist on your system. The headers for 4.18.0-15-generic exist, but they are for a *different* kernel version.

This is a common issue in Labtainer or virtual environments where the running kernel might be updated, but the corresponding headers aren't, or a specific kernel version is used for the lab.

**Solution: Point Makefile to the Correct (Existing) Kernel Headers**

Since you can't install headers for 6.11.0-29-generic via apt, and the lab environment likely has a fixed set of headers, the solution is to **modify your Makefile to point to the kernel headers that *actually exist* on your system.**

From your ls -l /usr/src/ output, the available headers are:

* /usr/src/linux-headers-4.18.0-15
* /usr/src/linux-headers-4.18.0-15-generic (which seems to be a set of symlinks to the first one)

We should use the 4.18.0-15-generic path as it's the more specific one usually used for module builds.

**Step 1: Edit your Makefile**

1. Open your Makefile for editing:

Bash

nano Makefile

1. Find the line that defines KDIR:

Makefile

KDIR := /lib/modules/$(shell uname -r)/build

1. **Change this line** to directly point to the existing kernel headers:

Makefile

KDIR := /usr/src/linux-headers-4.18.0-15-generic

This bypasses the uname -r command in the Makefile and explicitly tells make where to find the headers.

1. Save the Makefile: Press Ctrl+O, then Enter to confirm, and Ctrl+X to exit nano.

**Step 2: Ensure build-essential is installed**

Just to be absolutely sure, run this command again, as it provides the core make and gcc tools:

Bash

sudo apt update

sudo apt install build-essential

It should say that build-essential is already the newest version, or install it if it somehow got missed.

* + Quá trình này sẽ sử dụng Makefile để biên dịch mã nguồn C thành một mô-đun nhân Linux (Linux Kernel Module - LKM) có phần mở rộng .ko (kernel object).

A screenshot of a computer

AI-generated content may be incorrect.

1. Nếu biên dịch thành công, một file có tên như **lkmdemo.ko** sẽ xuất hiện trong cùng thư mục.

**d. Tạo và ẩn file kiểm tra**

1. Tạo một file bất kỳ mà bạn muốn rootkit ẩn đi. Ví dụ, tạo file hidden\_file.txt với nội dung tùy ý:

nano hidden\_file.txt

Nhập nội dung bất kỳ, sau đó lưu lại (Ctrl+O, Enter, Ctrl+X).

**e. Chèn Rootkit vào Kernel**

1. Sử dụng lệnh insmod với quyền sudo để chèn mô-đun lkmdemo.ko vào nhân Linux:

sudo insmod lkmdemo.ko

1. Nếu chèn thành công, có thể không có thông báo đặc biệt, nhưng mô-đun đã được tải.

**f. Kiểm tra hoạt động của Rootkit**

1. Kiểm tra xem file đã được giấu thành công và mô-đun đã được chèn vào kernel hay chưa:

Bash

ls && cat hidden\_file.txt && lsmod | grep lkmdemo > output.txt

* + **ls**: Lệnh này sẽ liệt kê các file và thư mục trong thư mục hiện tại. Nếu rootkit hoạt động, bạn sẽ **không thấy hidden\_file.txt** trong danh sách này.
  + **cat hidden\_file.txt**: Mặc dù không thấy file trong ls, lệnh cat vẫn có thể **đọc được nội dung của file đó**. Điều này chứng tỏ file vẫn tồn tại trên hệ thống nhưng đã bị rootkit che giấu khỏi các công cụ thông thường.
  + **lsmod | grep lkmdemo**: Lệnh này sẽ hiển thị danh sách các mô-đun nhân đang tải và lọc ra mô-đun lkmdemo. Nếu có kết quả, điều đó xác nhận mô-đun rootkit của bạn đang chạy trong kernel.

1. Phân tích file output.txt để xác nhận kết quả.
   * **Kết quả mong đợi**: Lệnh ls không hiển thị hidden\_file.txt, nhưng lệnh cat hidden\_file.txt vẫn hiển thị nội dung của file. Lệnh lsmod | grep lkmdemo hiển thị thông tin về mô-đun lkmdemo. Đây là bằng chứng rõ ràng cho thấy rootkit đã hoạt động thành công.

**g. Gỡ bỏ Rootkit khỏi Kernel**

1. Để gỡ bỏ mô-đun rootkit khỏi kernel, sử dụng lệnh rmmod:

Bash

sudo rmmod lkmdemo.ko

1. Sau khi gỡ bỏ, bạn có thể kiểm tra lại bằng ls để xem file đã ẩn có xuất hiện trở lại hay không, và lsmod | grep lkmdemo để xác nhận mô-đun đã bị gỡ.

**4. Kết thúc và Khởi động lại Lab**

**a. Kết thúc bài Lab:**

1. Trên Terminal đầu tiên (Terminal dùng để khởi động Labtainer), sử dụng câu lệnh sau để kết thúc bài lab:

Bash

stoplab pen\_attk\_mitre\_t1014

1. Khi bài lab kết thúc, một tệp .zip lưu kết quả sẽ được tạo và lưu vào một vị trí được hiển thị bên dưới lệnh stoplab.

**b. Khởi động lại bài Lab (nếu cần):**

1. Nếu trong quá trình làm bài bạn cần thực hiện lại từ đầu, sử dụng câu lệnh:

Bash

startlab -r pen\_attk\_mitre\_t1014

**5. Kết luận và Nhận thức bảo mật**

Thông qua bài thực hành này, sinh viên đã trực tiếp trải nghiệm cách một **rootkit** có thể thao túng nhân Linux để che giấu sự hiện diện của các tệp độc hại. Việc này giúp củng cố kiến thức về:

* **Tính nguy hiểm của rootkit**: Khả năng ẩn mình sâu trong hệ điều hành khiến chúng cực kỳ khó bị phát hiện và loại bỏ. Chúng cho phép kẻ tấn công duy trì quyền kiểm soát hệ thống trong thời gian dài mà không bị phát hiện.
* **Cơ chế hoạt động của Kernel Module**: Bài thực hành minh họa cách các mô-đun nhân có thể được tải vào kernel và thay đổi hành vi của hệ thống ở cấp độ thấp.
* **Tầm quan trọng của giám sát hệ thống**: Để phát hiện rootkit, cần có các công cụ chuyên dụng (ví dụ: rkhunter, chkrootkit) hoặc phân tích sâu hành vi hệ thống để tìm kiếm các dấu hiệu bất thường mà rootkit không thể che giấu hoàn toàn.
* **Phòng chống**: Luôn cập nhật hệ thống, sử dụng phần mềm bảo mật, kiểm tra tính toàn vẹn của file hệ thống, và hạn chế quyền root là những biện pháp quan trọng để ngăn chặn rootkit.

Bài thực hành này không chỉ trang bị kiến thức kỹ thuật mà còn nâng cao nhận thức về các mối đe dọa bảo mật tinh vi trong thế giới thực, giúp sinh viên chuẩn bị tốt hơn cho vai trò bảo vệ hệ thống thông tin.