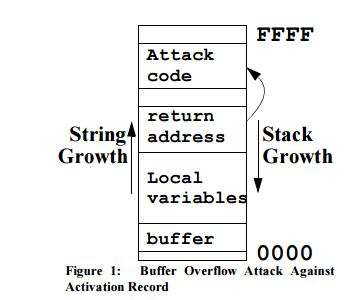
1. Giới thiệu

Khai thác lỗi tràn bộ đệm được biết đến đầu năm 1988, và đến năm 1996 đã có bài hướng dẫn từng bước cho việc khai thác các lỗ hổng tràn bộ đệm trên stack. Lỗi tràn bộ đệm là khi bộ nhớ bị ghi đè nhiều lần trên ngăn xếp. Thông thường nó xảy ra do người dùng gởi một lượng lớn dữ liệu đến server ứng dụng, kết quả là dữ liệu đó sẽ đè lên các vị trí bộ nhớ liên kề.

Tấn công buffer overflow có nguyên nhân gần giống với tấn SQL Injection , khi người dùng hay hacker cung cấp các biên đầu vào hay dữ liệu vượt quá khả năng xử lý của chương trình làm cho hệ thống bị treo, dẫn tới từ chối dịch vụ hay có khả năng bị các hacker lợi dựng chèn các chỉ thị trái phép nhằm thực hiện các đoạn mã nguy hiểm từ xa. Nguyên nhân chính có thể là do:

* + Phương thức kiểm tra biên không được thực hiện đầy đủ hoặc là được bỏ qua.
  + Các ngôn ngữ lập trình, như C bản thân nó đã tiềm ẩn các lỗi mà hacker có thể khai thác. Trong C còn có các hàm không kiểm tra những buffer được cấp phát trên stack có kích thước lớn hơn dữ liệu được copy và bộ đệm hay không.



1. Nội dung

2.1 Các kiểu khai thác lỗi buffer overlfow

* + Khai thác lỗi tràn bộ đệm trên stack:
    - Ghi đè một biến địa phương nằm gần bộ đệm trong stack để thay đổi hành vi của chương trình nhằm tạo thuận lợi cho kẻ tấn công
    - Ghi đè địa chỉ trả về trong một khung stack. Khi hàm trả về, thực thi sẽ được tiếp tục tại địa chỉ mà kẻ tấn công đã chỉ rõ, thường là tại một bộ đệm chứa dữ liệu vào người dùng

Ví dụ: khi dữ liệu tràn xuống stack của một hàm (function) đang chạy thì dữ liệu có thể đè vào địa chỉ trả về (return address) của hàm làm cho nó trỏ đi chỗ khác. Kẻ tấn công (Intruder) có thể bỏ một đoạn mã vào “chỗ khác” này để phá phách, truy cập dữ liệu, vv... (chỗ khác này thường là nằm ở chính bộ đệm đang bị tràn.)

* + Khai thác lỗi tràn bộ đệm trên heap
    - Một hiện tượng tràn bộ đệm xảy ra tỏng khu vực dữ liệu heap được gọi là hiện tượng tràn heap và có thể khai thác được bằng các kỹ thuật khác với các lỗi trần stack
    - Bộ nhớ heap được cấp phát động bởi các ứng dụng thời gian chạy và thường chứa dữ liệu của chương trình.
    - Việc khai thác được thực hiện bằng cách phá dữ liệu này theo các cách đặc biệt để làm cho ứng dụng ghi đè lên các cấu trúc ́dữ liệu nội bộ, chẳng hạn các con trỏ của danh sách liên kết.
  + Một số cách khai thác khác:
    - Khai thác dựa vào các lỗ hổng phần mềm thông qua ngôn ngữ lập trình (phần mềm thương được viết bằng ngôn ngữ C).
    - Khai thác các trang web có tương tác người dùng nhưng không ràng buộc dữ liệu nhập như username, password...

2.2 Cách phát hiện lỗi buffer overflow:

* + Bạn thử copy 1 số nguyên, nhưng thay vì bạn gửi 1 lượng lớn dữ liệu để nó chiếm không gian bộ nhớ gấp đôi bộ nhớ dành cho số nguyên. Với việc copy tùy ý, lượng dữ liệu lớn này sẽ được copy vào khu vực kế tiếp và được đặt trong ngăn xếp. Đôi khi dữ liệu này được lôi ra khỏi ngăn xếp và thực thi. Điều này có thể ngay lập tức làm hỏng hóc vùng nhớ dữ liệu khác. Lỗi tràn bộ đệm được sử dụng để tấn công vào máy chủ giành quyền kiểm soát mà bình thường không hề xảy ra hoặc nó khởi động chương trình từ ngăn xếp.
  + Một trình biên dịch mà hiểu các thuộc tính của thư viện thời gian tiêu chuẩn sẽ biết rằng bộ đệm bây giờ chứa dữ liệu từ một thời điểm không xác định trong hệ thống tập tin bên ngoài chương trình và có khả năng bị nguy hiểm
  + Sự dụng phân tích mã nguồn để tìm lỗi tràn bộ đệm trong mã thực thi một các chính xác đòi hỏi phải hiểu biết sâu về code. Và công nghệ của trình biên dịch được thiết kế với mục đích tìm kiếm lỗi tràn bộ đệm hiệu quả và chính xác hơn các phương pháp khác.

2.3 Biện pháp ngăn chặn

Việc xử lý trước khi đọc có thể làm giảm bớt lỗi tràn bộ đêm nhưng không thể ngăn chặn được một cách tuyệt đối. Xử lý bao gồm:

* + Chuyển từ chữ hoa thành chữ thường
  + Loại bỏ các ký tự đặc biệt và lọc các xâu không chứa kí tự là chữ số hoặc chữ cái.
  + Ngoài ra vẫn còn có các kỹ thuật để tránh việc lọc và xử lý này:
    - Alphanumeric code: Mã gồm toàn chữ và số.
    - Polumorphic code: mã đa hình.
    - Seft-nodifying code: mã tự sửa đội.
    - Tấn công kiểu return - to - libc

Vì vậy, để tránh các nguy cơ bị khai thác lỗi buffer overflow chúng ta cần sử dụng các biện pháp phòng tránh hiệu quả hơn:

* Lựa chọn ngôn ngữ lập trình: ngôn ngữ lập trình có một ảnh hưởng lớn đối với sự xuất hiện lỗi tràn bộ đệm
* Sử dụng các thư viện an toàn: các thư viện đưuọc viết tốt và đã được kiểm thử dành cho các kiểu dữ liệu trừu tượng mà các thư viện này thực hiện tự động việc quản lý bộ nhớ
* Để bảo vệ chống tràn bộ đệm, việc xác nhận đầu vào là rất cần thiết. Bất kỳ field, form đăng ký, hoặc bất kỳ phương pháp khác mà có nhập đầu vào đều có thể có một lỗ hổng tiềm tàng của việc tràn bộ đệm. Điều này không chỉ bao gồm một form field, HTTP header, giao diện XML, hoặc bất kỳ giá trị đầu vào nào khác. Bạn hãy nghĩ đến tất cả những giá trị input ở bất kì khu vực nào của ứng dụng phần mềm.

=> Một số trong những điều đơn giản hơn là dev cần thực hiện thiết lập kích thước giới hạn cho phép của các giá trị đầu vào. Mặc dù ai đó có thể có một cái tên với 100 kí tự trong đó, giới hạn kích thước của các field nhập dữ liệu. Bạn cần phải biết chính xác kích thước của dữ liệu tiềm năng này có thể đạt đến. Điều này có thể đơn giản như setting thẻ HTML trong input field, hoặc sử dụng một số javascript để kiểm tra và validate input field. Nhưng việc được cảnh báo như vậy là chưa đủ. Một hacker sẽ không tuân theo việc sử dụng trình duyệt mà sẽ tự động set kích thước giá trị đầu vào và sử dụng javascript validate input. Một hacker có thể và sẽ sử dụng cái gì khác. Một hacker sẽ viết code của chính anh ấy, sử dụng netcat, hoặc một số hình thức khác của một trình duyệt để có được xung quanh những kiểm tra đơn giản.