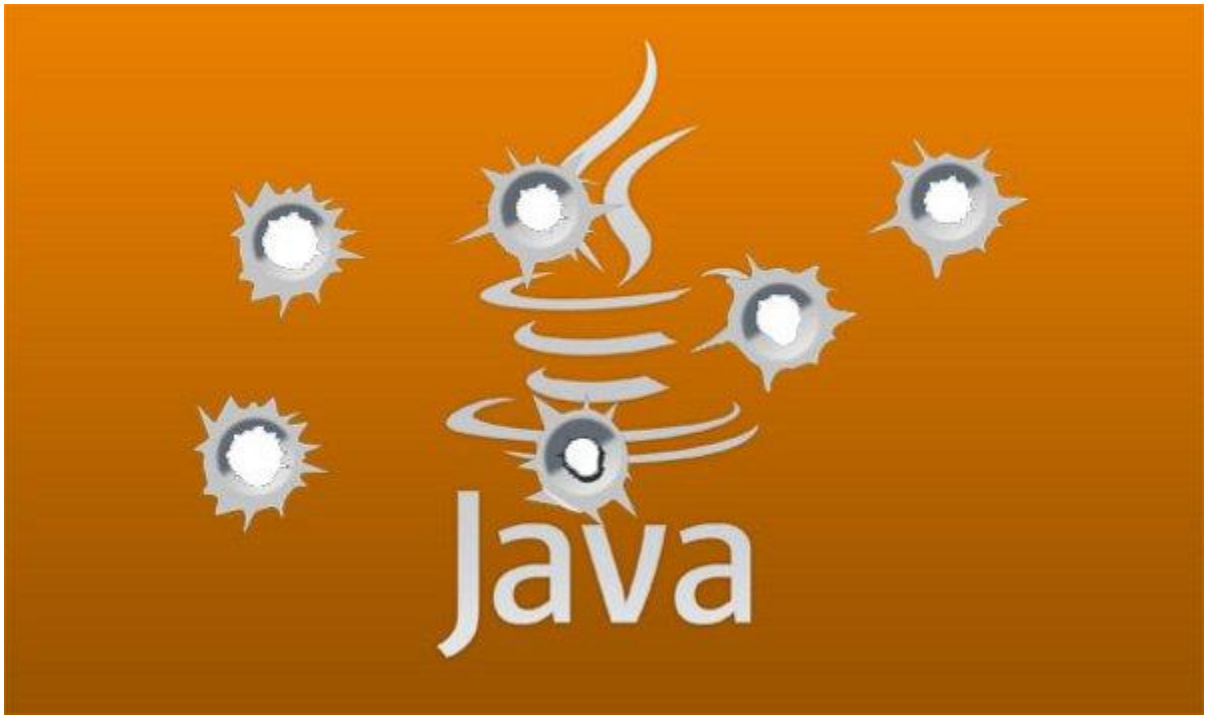
Lỗ hổng bảo mật - những hiểu biết căn bản

Lỗ hổng phần mềm có thể được hiểu như là một trục trặc hoặc điểm yếu trong phần mềm hoặc trong hệ điều hành. Với sự phát triển của các công nghệ tấn công mới, mức độ nghiêm trọng của lỗ hổng phần mềm đang tăng lên theo cấp số nhân. Tất nhiên, tất cả các hệ thống đều ẩn chứa những lỗ hổng, nhưng vấn đề nằm ở chỗ liệu những lỗ hổng này có bị khai thác và gây ra những thiệt hại hay không.
Các thảm họa an ninh mạng gây ra bởi các lỗ hổng phần mềm thường được giải thích bởi ba yếu tố lý tưởng. Đó là:

- Sự tồn tại: Đó là sự tồn tại của một lỗ hổng trong phần mềm.
- Sự truy cập: Đó là khả năng mà tin tặc có thể truy cập vào một lỗ hổng bảo mật.
- Sự Khai thác: Đó là khả năng mà hacker có thể tận dụng và thu về lợi nhuận từ lỗ hổng đó thông qua các công cụ hoặc với một số kỹ thuật nhất định.

Ngày nay, rất nhiều các tổ chức đang phải chứng kiến những lỗ hổng trong hệ thống của họ bị khai thác. Ví dụ, dưới đây là bảng hiển thị 10 sản phẩm phần mềm hàng đầu có nhiều lỗ hổng bảo mật nhất trong năm 2016 theo tham chiếu của hệ thống CVSS:

Số thứ tự	Tên phần mềm	Nhà phát triển	Số lỗ hổng
1	Android	Google	523
2	Debian Linux	Debian	327
3	Ubuntu Linux	Ubuntu	278
4	Flash Player	Adobe	266
5	Leap	Novell	260
6	Opensuse	Novell	228
7	Acrobat Reader Dc	Adobe	277
8	Acrobat Dc	Adobe	277
9	Acrobat	Adobe	224
10	Linux Kernel	Linux	217



Lỗi phần mềm

Ngay cả những phần mềm tầm trung đơn giản, chỉ phục vụ một vài tác vụ chuyên biệt cũng đã tạo thành từ một lượng lớn code. Cấu trúc phần mềm được thiết kế bởi con người, và những dòng code trong đó cũng được viết bởi con người, vì vậy việc xuất hiện lỗi là không thể tránh khỏi. Trong phần lớn trường hợp, nếu một phần mềm được sản xuất một cách chuyên nghiệp – các lỗi này không thể có tác động gì quá lớn, nhất là đến các khía cạnh về bảo mật. Cùng lắm ta sẽ thấy một vài chức năng không hoạt động, đôi lúc phần mềm *"treo"* khi đang làm việc hoặc làm việc chậm chạp...

Nhưng nói vậy không có nghĩa là những lỗi nghiêm trọng liên quan đến bảo mật không thể xảy ra. Nói cụ thể hơn một chút, đó là những lỗi phần mềm mà người ngoài có thể khai thác để tác động thay đổi cách phần mềm vận hành, đưa thêm vào các đoạn mã tự viết, xem các dữ liệu mà phần mềm quản lí... Ngoài các nguyên nhân chủ quan như sự bất cẩn khi sử dụng của người dùng (click vào đường link lạ, download các phần mềm độc hại), các lỗi này là một trong những khe hở chính mà tin tặc thường tập trung khai thác để xâm nhập vào các hệ thống máy móc – từ các máy chủ đến các máy cá nhân của người dùng cuối. Nếu lỗ hổng này thuộc về một phần mềm không phổ biến, chỉ phục vụ vài tác vụ đơn giản và không có vai trò quan trọng trong hệ thống, hiển nhiên hiểm họa về bảo mật vẫn có nhưng không nghiêm trọng. Nhưng hệ thống phần mềm càng phức tạp, đồ sộ thì hiển nhiên việc kiểm soát sự xuất hiện của những lỗi này càng khó – bất kể các kĩ sư thiết kế có trình độ cao đến đâu. Và chính những phần mềm này lại thường chiếm vai trò chủ chốt, cũng như tác động đến nhiều ngóc ngách của hệ thống. Nhờ len lỏi qua kẽ hở tạo ra bởi lỗi của những phần mềm này, kẻ xấu có thể thực hiện những thay đổi nhất định lên máy móc của người dùng, hay nắm được quyền điều khiển, truy cập các thông tin nhạy cảm.

5 lỗ hổng bảo mật quan trọng và cách thức tấn công

Để xây dựng phần mềm an toàn, không thể thiếu được sự hiểu biết về các lỗ hổng phần mềm. Ở đây, chúng chúng ta sẽ tìm hiểu một cách tổng quan ngắn gọn về các lỗ hổng bảo mật quan trọng và nguy hiểm.

SQL injection

Các lỗ hổng SQL injection mang lại cơ hội để các hacker chèn mã độc vào một câu lệnh SQL. SQL Injection là một trong những kiểu hack web bằng cách inject các

mã SQLquery/command vào input trước khi chuyển cho ứng dụng web xử lí, bạn có thể login mà không cần username và password, remote execution (thực thi từ xa), dump data và lấy root của SQL server. Công cụ dùng để tấn công là một trình duyệt web bất kì, chẳng hạn như Internet Explorer, Netscape, Lynx…

Vị trí trong danh sách liệt kê lỗ hổng (CWE)

Xếp hạng	ID	Tên
1	CWE-89	"Thất bại trong duy trì cấu trúc truy vấn SQL (còn được gọi là SQL injection)"

Những ngôn ngữ lập trình bị ảnh hưởng

Bất kỳ ngôn ngữ mã hóa nào có thể được sử dụng trực tiếp với cơ sở dữ liệu SQL đều dễ bị tổn thương bởi kiểu tấn công này. uy nhiên, dưới đây là một số ngôn ngữ phổ biến nhất:

- Các ngôn ngữ cấp cao: Perl, Ruby, Python, Java, VB.Net, SQL
- Server Page: ASP, JSP, ASP.NET, PHP
- Các ngôn ngữ cấp thấp hơn: C, C++

OS Command Injection

Các lỗ hổng OS Command Injection xảy ra khi phần mềm tích hợp dữ liệu do người dùng quản lý trong một lệnh, các dữ liệu này được xử lý trong trình thông dịch lệnh. Nếu dữ liệu không được kiểm tra, một hacker có thể sử dụng các siêu ký tự shell để thay đổi lệnh đang được thực thi.

Vị trí trong danh sách liệt kê lỗ hổng (CWE)

Xếp hạng	ID	Tên
2	CWE 78	"OS Command Injection"

Buffer Overflow

Buffer overflow (tràn bộ nhớ đệm) là một loại lỗ hổng bảo mật nổi tiếng. Nó xảy ra khi một chương trình cố gắng load thêm nhiều dữ liệu vào bộ đệm, vượt quá dung lượng lưu trữ cho phép của nó.Việc dữ liệu được ghi bên ngoài có thể làm hỏng chương trình, hỏng dữ liệu và thậm chí tạo điều kiện cho việc thực thi các mã độc. Các ngôn ngữ như Java, Python, Visual Basic và C # bao gồm các mảng kiểm tra ràng buộc và các kiểu chuỗi gốc. Do đó, tràn bộ đệm được coi là không thể xảy ra trong những môi trường được viết bằng các ngôn ngữ này.

Vị trí trong danh sách liệt kê lỗ hổng (CWE)

Xếp hạng	ID	Tên

3	CWE-120	“Classic Buffer Overflow”

Bảng sau đây cho thấy các mục có liên quan của lỗ hổng bảo mật này trong danh sách CWE:

ID	Tên
CWE 121	“Stack-based Buffer Overflow“
CWE 122	“Heap-based Buffer Overflow”
CWE 123	“Write-what-where Condition”
CWE 124	“Boundary Beginning Violation”
CWE 125	“Out of bounds Read”
CWE 128	“Wrap around Error”
CWE 129	“Unchecked Array Indexing”
CWE 131	“Incorrect Calculation of Buffer Size”
CWE 193	“Off by One Error”
CWE 466	“Return of Pointer Value Outside of Expected Range”

Những ngôn ngữ lập trình bị ảnh hưởng

- Ngôn ngữ: C, Fortran, Assemblys.
- Môi trường: Các máy chủ của ứng dụng, máy chủ web và ứng dụng web.

Uncontrolled Format String

Lỗ hổng này bao gồm việc chấp nhận kết quả đầu vào không được kiểm soát hoặc trái phép dưới dạng chuỗi định dạng để thực thi một hàm. Điểm yếu này có thể dẫn đến việc thực thi các mã độc hại và thậm chí có thể làm hỏng hệ thống.

Vị trí trong danh sách liệt kê lỗ hổng (CWE)

Xếp hạng	**ID**	**Tên**
23	CWE 134	"Uncontrolled Format String"

Những ngôn ngữ lập trình bị ảnh hưởng

- Ảnh hưởng Trực tiếp: C, C ++.
- Ảnh hưởng gián tiếp: Perl (nếu đọc trong một loại dữ liệu giả mạo).

Integer Overflow

Lỗ hổng integer overflow (tràn số nguyên) tồn tại khi một phép tính cố gắng tăng giá trị số nguyên cao hơn số nguyên được sử dụng để lưu trữ trong biểu thức có liên quan. Khi lỗi này xảy ra, giá trị số nguyên có thể chuyển đổi thành số âm hoặc rất nhỏ. Điểm yếu này trở thành một vấn đề bảo mật quan trọng khi kết quả tính toán được sử dụng để xử lý vòng lặp điều khiển, xác định kích thước hoặc thực thi các nhiệm vụ như sao chép, cấp phát bộ nhớ, ghép nối… và đưa ra quyết định.

Vị trí trong danh sách liệt kê lỗ hổng (CWE)

Xếp hạng	**ID**	**Tên**
24	CWE 190	"Integer Wraparound or Overflow"

Bảng sau đây cho thấy các mục có liên quan của lỗ hổng bảo mật này trong danh sách CWE:

ID	**Tên**
CWE 682	"Incorrect Calculation "

CWE 191	“Integer Underflow”
CWE 192	“Coercion Error”

Những ngôn ngữ lập trình bị ảnh hưởng

Hầu hết tất cả các ngôn ngữ đều bị ảnh hưởng; tuy nhiên, các hậu quả xảy ra cũng sẽ khác nhau tùy thuộc vào cách ngôn ngữ đó xử lý các số nguyên.

- Bị ảnh hưởng nghiêm trọng: C, C++

Zero-Day Exploits – Đòn tấn công âm thầm

Thực tế, các lỗ hổng có thể bị khai thác sử dụng cho mục đích xấu tồn tại trên bất cứ phần mềm nào. Thậm chí có những phần của thiết kế khó có thể bị cho là lỗi cho đến khi xuất hiện những công nghệ cho phép người ngoài khai thác nó – khiến cho tác giả phải thiết kế lại cách sản phẩm của mình vận hành. Khi cập nhật phần mềm mới, ngoài việc đôi lúc thấy xuất hiện các chức năng mới, hay hiệu năng hoạt động được cải thiện, chắc hẳn không ít lần bạn thấy changelog (danh sách các thay đổi) xuất hiện một loạt các sửa chữa lỗi gần đây nhất. Những người tạo ra một sản phẩm dĩ nhiên phải là người hiểu rõ đứa con cưng của mình nhất – và sẽ cố hết sức để sửa chữa lỗi mỗi khi phát hiện ra (ít nhất thì phần lớn trường hợp là như vậy). Với sản phẩm phổ biến trên thị trường, được phát hành bởi các công ty- tổ chức hoạt động một cách chuyên nghiệp, điều này càng đúng hơn.

Nhưng không có gì là tuyệt đối. Sẽ có những lúc mà tác giả phát hiện lỗi sau người ngoài, hoặc thậm chí là không đủ khả năng phát hiện ra. Không phải bỗng nhiên mà các hãng lớn thường tổ chức những cuộc thi về khai thác lỗ hổng trên sản phẩm của mình, đồng thời tuyển mộ nhân lực từ các cuộc thi đó, cũng như tuyển mộ các tin tặc hoàn lương. Thực tế vẫn luôn như vậy: có người có tài, có người không. Thậm chí sẽ có những lúc hãng sản xuất phát hiện lỗi, nhưng thời gian để hoàn thành việc sửa chữa lại lâu hơn thời gian tin tặc cần để viết ra công cụ khai thác, đồng thời hoàn thành công việc phá hoại, gián điệp hay trộm cắp bằng công cụ đó. Đó cũng là một trong những lí do khiến ta thấy các bài viết về lỗ hổng bảo mật thường chỉ xuất hiện nhiều tháng sau khi lỗi đã được sửa. Các hacker mũ trắng quá hiểu rằng việc sửa lỗi đôi lúc khó khăn và phức tạp hơn nhiều lần so với việc lợi dụng lỗi cho mục đích xấu, vì vậy họ thường cho hãng sản xuất hàng tháng trời để sửa chữa sai lầm của mình trước khi công bố chi tiết về lỗ hổng mà mình phát hiện ra ngoài để phục vụ mục đích nghiên cứu.

Còn kịch bản xấu nhất? Kẻ xấu phát hiện ra lỗi... và dĩ nhiên là không công bố cho ai biết, âm thầm đóng cửa tu luyện để hoàn thành công cụ khai thác lỗi và âm thầm phát tán (thường thấy nhất là dưới dạng virus, worm,trojan…). Thậm chí giới tội phạm có thể đem những thông tin này ra giao dịch, trao đổi ngầm với nhau, hay bán kèm trong những bộ kit được viết ra chuyên để phục vụ việc tìm hiểu, khai thác lỗ hổng. Hãng sản xuất hoàn toàn không biết sự tồn tại của lỗ hổng đó chứ đừng nói đến việc tìm cách sửa. Chỉ đến khi hậu quả đã sờ sờ ra trước mắt, họ mới

có thể tá hỏa lên tìm cách khắc phục, đền bù cho người dùng, như vụ việc của Sony ngày trước. Cũng chính vì đòn tấn công được thực hiện khi hãng sản xuất hoàn toàn chưa biết đến sự tồn tại của các lỗ hổng này, có *"0 ngày"* để tìm cách vá lỗi mà cái tên *"zero-day"* ra đời.

Tóm lại, việc một lỗi phần mềm tồn tại vốn không phải việc gì quá kì lạ, hiểm họa chỉ xuất hiện khi hãng sản xuất thua trong cả 2 cuộc đua: phát hiện lỗi và sửa lỗi.

Quá trình khai thác

Cần hiểu rằng, các công cụ về bảo mật hiện đại ngày nay như tường lửa, phần mềm anti-virus, anti-malware… thường có cơ chế hoạt động thông minh để phát hiện khi một đoạn mã nào đó có hành vi đáng ngờ, bất kể đoạn mã đó có sẵn trong cơ sở dữ liệu về virus, malware hay không. Cũng tương tự như một trinh sát dày dạn có thể phát hiện dấu hiệu khả nghi của một kẻ trộm mà không cần lệnh truy nã hay chữ *"trộm"* to đùng trước trán. Tuy vậy như đã nói, trường hợp xấu nhất là khi các tin tặc phát hiện lỗi chưa ai biết tới, viết một công cụ hoàn toàn mới để khai thác. Một kẻ nếu đủ khả năng để về đích đầu tiên trong cả 2 cuộc đua này (ở đây không nói đến những đối tượng sử dụng lại công cụ) hẳn nhiên thừa kinh nghiệm trong việc tránh ánh mắt dò xét của các công cụ bảo mật. Vì vậy cho đến khi lỗ hổng hoàn toàn được vá, mọi biện pháp mà các công cụ bảo mật cung cấp đều chỉ mang tính tạm thời. Chuỗi sự kiện điển hình thường là như sau:

1. Xuất hiện một lỗ hổng có thể bị khai thác bằng các công nghệ hiện có.

2. Kẻ tấn công phát hiện lỗ hổng.

3. Kẻ này lập tức tiến hành viết và phát tán công cụ khai thác lỗ hổng này.

4. Hãng sản xuất đồng thời phát hiện lỗi và lập tức tìm cách sửa chữa.

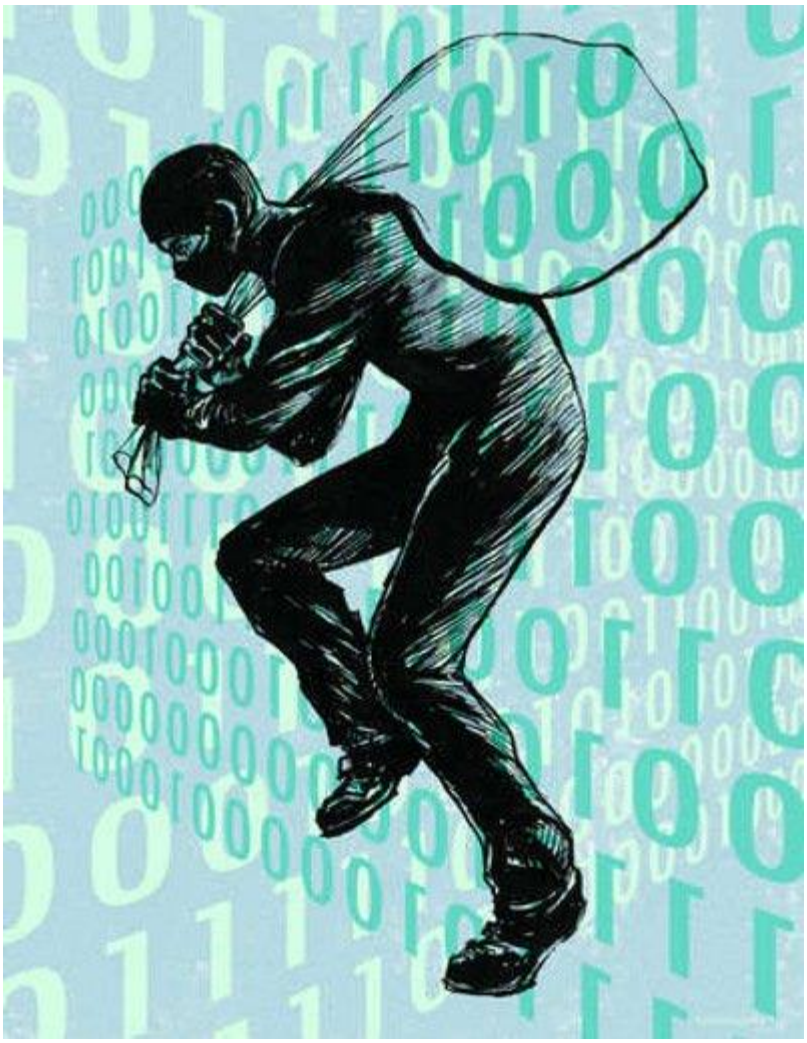
5. Lỗ hổng được công bố ra ngoài.

6. Các phần mềm anti-virus được cập nhật thông tin để phát hiện khi có các đoạn mã tìm cách khai thác lỗ hổng này.

7. Hãng sản xuất hoàn thành bản vá.

8. Hãng hoàn tất phát hành bản vá lỗi đến tất cả khách hàng.

Thời điểm của đợt tấn công đầu tiênt hiển nhiên nằm giữa bước 3 và 5. Theo một nghiên cứu mới đây của đại học Carnegie Mellon của Mỹ, giai đoạn này trung bình kéo dài 10 tháng. Tuy nhiên không phải lúc nào tất cả người dùng cuối cũng bị nguy hiểm trong giai đoạn này. Dạng tấn công tận dụng thời điểm hãng sản xuất chưa phát hiện (hoặc chưa sửa được lỗi) này có lợi thế lớn nhất là sự kín đáo – phù hợp cho việc lấy trộm thông tin hoặc phá hoại ngầm mà không bị phát hiện. Vì vậy giai đoạn này đối tượng bị nhắm đến thường là một nhóm người có thể đem lại lợi ích cụ thể cho kẻ tấn công để sau đó hắn có thể rút đi êm thấm. Mục tiêu dó có thể là các tổ chức, tập đoàn mà kẻ này muốn phá hoại hoặc các thông tin tài khoản có thể sử dụng để kiếm lời.



Cũng theo nghiên cứu này, giai đoạn từ bước 5 đến 8 mới thực sự nguy hiểm. Đây là lúc thông tin về lỗ hổng được công bố, và cùng với các công ty phát triển anti-

virus, những tin tặc chưa biết đến lỗi này cũng có thể tiếp cận được thông tin. Làn sóng tấn công lúc này không còn âm thầm, mà dồn dập hơn rất nhiều. Nếu ví đợt tấn công trước đó nguy hiểm như một nhát dao đâm sau lưng, thì đợt tấn công lúc này như một chuỗi đòn đánh trực diện, không hiệu quả với những ai cẩn thận đề phòng nhưng vẫn không kém phần nguy hiểm nếu như gặp đúng những người lơ là bảo mật hoặc nhỡ sử dụng công cụ bảo mật kém chất lượng, cập nhật chậm. Những đối tượng không có khả năng phát hiện lỗi, cũng như không có khả năng phát triển công cụ cũng tham gia từ thời điểm này, khiến việc phát tán và tìm đến những cỗ máy có hệ thống bảo mật yếu kém nhanh hơn rất nhiều. Khi số lượng kẻ tham gia tấn công tăng lên, động cơ và phương thức tấn công cũng đa dạng hơn chứ không thể chỉ thuần túy là len lỏi và trộm cắp nữa.

Sau khi đọc đến đây, chắc bạn đọc cũng hiểu rằng, khi nói đến việc bảo vệ thông tin và hệ thống của mình, ngoài việc cập nhật các biện pháp phòng thủ thì việc cập nhật thông tin cũng quan trọng không kém. Thường thì những lỗi nghiêm trọng của những hệ thống phổ biến và quan trọng như Java vừa qua sẽ được báo chí đăng tải nhan nhản ngay khi hãng sản xuất công bố. Tuy nhiên những phần mềm có danh tiếng và độ phổ biến *"khiêm tốn"* hơn thì thường không được ưu ái như vậy. Vì vậy ngoài việc chú ý nâng cấp bản vá lỗi, cần dừng việc sử dụng những phần mềm cũ kĩ không còn được chăm sóc, sửa lỗi ngay khi có thể. Ví dụ? Microsoft vẫn không ngừng kêu gào để những XP, IE6 được yên nghỉ đấy thôi...