**File Upload vulnerabilities**

**Khái niệm:**

- Lỗ hổng tải tập tin là khi máy chủ cho phép người dùng tải tập tin lên hệ thống mà không xác thực tên, loại, nội dung và kích thước của chúng.

- Nếu không hạn chế được, một chức năng đơn giản như tải hình ảnh cũng có thể tải những tập tin nguy hiểm như kịch bản phía máy chủ để kích hoạt mã từ xa.

**Những tác động của lỗ hổng:**

- Phụ thuộc vào 2 yếu tố chính:

* Ứng dụng không xác thực đầy đủ tập tin: kích thước, loại, nội dung,…
* Những hạn chế khi tải tập tin

- Trong trường hợp tệ nhất, loại tập tin không được xác thực đúng và máy chủ cho phép những loại tập tin như .php hay .jsp được kích hoạt. Kẻ tấn công có thể tải đoạn mã phía máy chủ như web shell, giành quyền kiểm soát toàn máy chủ.

- Nếu tên tập tin không được xác thực, kẻ tấn công có thể ghi đè những tập tin quan trọng bằng cách tải tập tin cùng tên. Nếu máy chủ bị dính lỗ hổng truyền file, kẻ tấn công có thể tải tập tin ở vị trí bất kì.

- Không đảm bảo kích thước tập tin được tải lên, có thể kích hoạt tấn công DOS khi kẻ tấn công có thể lấp đầy ổ đĩa trống.

**Lỗ hổng tải tập tin sinh ra như thế nào?**

Vì lỗ hổng này khá nguy hiểm, rất hiếm khi trang web không hạn chế tải tập tin. Nhưng việc xác thực mà nhà phát triển cho là mạnh mẽ lại rất dễ bị lỗi và vượt qua dễ dàng.

Ví dụ, bọn họ dùng danh sách đen những loại tập tin nguy hiểm, nhưng không phân tích rõ ràng khi kiểm tra🡪dễ bỏ xót những loại tập tin mờ mịt nhưng lại nguy hiểm.

Cuối cùng ,việc xác thực mạnh mẽ không được áp dụng đồng bộ qua mạng của các máy chủ, những thư mục hình thành trang web dẫn đến những điểm khác biệt có thể bị khai thác.

**Máy chủ web xử lí những yêu cầu cho tập tin tĩnh như thế nào?**

Máy chủ phân tích đường dẫn trong yêu cầu để xác định phần mở rộng tập tin. 🡪 So sánh nó với danh sách phần mở rộng đã được cấu hình .

* Nếu tập tin là loại không kích hoạt được như hình ảnh hay trang HTML tĩnh, máy chủ chỉ gửi nội dung tập tin đến máy khách trong phản hồi http.
* Nếu tập tin là loại kích hoạt được như PHP và máy chủ được cấu hình để kích hoạt loại này, nó sẽ gán những biến trên headers và tham số trong yêu cầu http và kết quả có thể được gửi trong phản hồi http.
* Nếu tập tin kích hoạt được nhưng không được máy chủ cấu hình, nó sẽ báo lỗi.

**Khai thác lỗ hổng bằng việc triển khai web shell:**

Từ góc độ bảo mật, viễn cảnh tệ nhất là trang web cho phép tải kịch bản phía máy chủ như PHP, Java hay Python và được cấu hình để kích hoạt nó.

🡪 Tạo 1 web shell trên máy chủ

Web shell là 1 tập lệnh độc hại cho phép kẻ tấn công thực hiện các lệnh bất kỳ trên máy chủ từ xa.

Nếu bạn tải web shell thành công, bạn có thể kiểm soát toàn máy chủ.

Bạn có thể đọc và ghi tập tin bất kì, trích xuất dữ liệu nhạy cảm, thậm chí dùng máy chủ để tấn công chống lại cơ sở hạ tầng bên trong và những máy chủ khác ngoài mạng.

Đoạn code PHP để đọc tập tin bất kì từ máy chủ:

<?php echo file\_get\_contents('/path/to/target/file'); ?>

Một web shell linh hoạt hơn cho phép bạn gửi 1 lệnh bất kì qua tham số :

<?php echo system($\_GET['command']); ?>

GET /example/exploit.php?command=id HTTP/1.1

**Khai thác lỗi xác thực khi tải tập tin:**

**Xác thực loại file bị lỗi:**

Khi gửi biểu mẫu HTML, trình duyệt gửi dữ liệu được cung cấp trong yêu cầu POST với loại dữ liệu là application/x-www-form-url-encoded. Điều này ổn khi gửi văn bản đơn giản nhưng không phù hợp với lượng lớn dữ liệu nhị phân như hình ảnh hay tài liệu pdf. Kiểu dữ liệu multipart/form-data sẽ phù hợp hơn.

Xem xét 1 biểu mẫu để tải hình ảnh, cung cấp mô tả và nhập họ tên người dùng.

POST /images HTTP/1.1

Host: normal-website.com

Content-Length: 12345

Content-Type: multipart/form-data; boundary=--------------------------012345678901234567890123456

---------------------------012345678901234567890123456

Content-Disposition: form-data; name="image"; filename="example.jpg"

Content-Type: image/jpeg

[...binary content of example.jpg...]

---------------------------012345678901234567890123456

Content-Disposition: form-data; name="description"

This is an interesting description of my image.

---------------------------012345678901234567890123456

Content-Disposition: form-data; name="username"

wiener

---------------------------012345678901234567890123456--

Nội dung tin được chia thành các phần riêng biệt cho từng đầu vào. Mỗi phần chứa Content-Disposition header cung cấp thông tin về đầu vào.

Mỗi phần cũng có thể chứa Content-Type header, báo cho máy chủ loại dữ liệu của đầu vào.

Ứng dụng xác thực việc tải tập tin bằng cách kiểm tra Content-type header có phù hợp với loại dữ liệu mà ứng dụng cho phép không. Vấn đề là ứng dụng tin vào dữ liệu mà người dùng kiểm soát được nên dễ bị bỏ qua việc xác thực.

**Ngăn chặn kích hoạt tập tin trong những thư mục mà người dùng truy suất được:**

Mặc dù rõ ràng tốt hơn là ngăn chặn những loại tập tin nguy hiểm được tải lên ngay từ đầu, tuyến phòng thủ thứ 2 là ngăn máy chủ thực thi bất kỳ tập lệnh nào lọt qua mạng.

Máy chủ chỉ chạy những tập lệnh mà MIME của nó được cấu hình rõ ràng. Ngược lại , nó sẽ báo lỗi hoặc trả về nội dung của tập tin ở văn bản thô.

GET /static/exploit.php?command=id HTTP/1.1

Host: normal-website.com

HTTP/1.1 200 OK

Content-Type: text/plain

Content-Length: 39

<?php echo system($\_GET['command']); ?>

**=>Điều này sẽ làm rò rỉ mã nguồn nhưng ngăn chặn những nỗ lực tạo web shell.**

Loại cấu hình này thường khác nhau giữa các thư mục. Ở những thư mục người dùng có thể tải tập tin thường kiểm soát chặt chẽ hơn những hệ thống tập tin người dùng không chạm được. Nếu bạn có thể tải những mã lệnh ở thư mục đó , lỗ hổng vẫn sẽ xảy ra bình thường 🡪 **Path Traversal**

**Lưu ý:** Máy chủ thường dùng mục filename ở những yêu cầu có kiểu nội dung là multipart/form-data để xác định tên và vị trí tập tin được lưu trữ.

**Danh sách đen không đủ các loại tập tin nguy hiểm:**

- Cách để ngăn chặn lỗ hổng này là tạo danh sách đen những tập tin có phần mở rộng nguy hiểm như .php.

- Cách này không hiệu quả vì không thể chặn hết những định dạng tập tin được như .php5 hay .shtml.

**Ghi đè những cấu hình của máy chủ:**

Máy chủ sẽ không kích hoạt tập tin nếu chúng không được cấu hình.

Ví dụ, trước khi máy chủ kích hoạt tập tin PHP được yêu cầu bởi máy khách, những nhà phát triển phải thêm những lệnh sau đến tập tin /etc/apache2/apache2.conf:

LoadModule php\_module /usr/lib/apache2/modules/libphp.so

AddType application/x-httpd-php .php

Rất nhiều máy chủ cho phép nhà phát triển tạo ra những tập tin cấu hình để ghi đè hoặc thêm cài đặt. Máy chủ apache sẽ tải 1 thư mục cấu hình từ tập tin .htaccess

Tương tự, những nhà phát triển có thể tạo những thư mục cấu hình trên máy chủ IIS bằng tập tin web.config. Nó có thể bao gồm những lệnh sau cho phép tập tin JSON hỗ trợ người dùng:

<staticContent>

<mimeMap fileExtension=".json" mimeType="application/json" /> </staticContent>

Máy chủ web dùng những tập tin cấu hình này nhưng bạn không được phép truy cập nó qua http. Tuy vậy, bạn vẫn thấy một số máy chủ cho phép tải lên những tập tin cấu hình độc hại. Trong trường hợp đó, nếu tập tin bạn cần trong danh sách đen, bạn có thể làm cho máy chủ kích hoạt nó.

**Làm xáo trộn phần mở rộng tệp**

Thậm chí danh sách đen đầy đủ nhất vẫn bị bỏ qua bằng kĩ thuật xáo trộn này.

- Xét 1 đoạn mã xác thực phân biệt hoa thường sẽ không nhận ra exploit.pHp chính là exploit.php.

- Kể cả không phân biệt hoa thường, điểm khác biệt vẫn cho phép bạn nén tập tin php độc hại để được kích hoạt bởi máy chủ.

- Bạn có thể thực hiện điều này bằng những kĩ thuật sau:

* Cung cấp nhiều phần mở rộng. Tập tin có thể được hiểu là php hoặc jpg: exploit.php.jpg
* Thêm kí tự ở cuối. Một vài thành phần sẽ bỏ qua khoảng trắng, dấu chấm ở cuối như exploit.php.
* Thử mã hóa URL (hoặc double URL) cho dấu chấm,’\’,’/’. Nếu giá trị không được giải mã ở phần xác thực nhưng sau đó được giải mã ở máy chủ, bạn có thể tải lên những tập tin độc hại: exploit%2Ephp
* Thêm dấu ; hoặc mã hóa URL kí tự null trước phần mở rộng tập tin. Nếu xác thực được viết ở ngôn ngữ cấp cao như PHP, Java nhưng máy chủ xử lí bằng ngôn ngữ cấp thấp trong C/C++: exploit.asp;.jpg hay exploit.asp%00.jpg

- Một số ứng dụng sẽ bỏ đi hay thay thế những phần mở rộng nguy hiểm để ngăn chặn tập tin được kích hoạt. Nếu nó không lặp lại, bạn có thể làm như sau: exploit.p.phphp🡪exploit.php

**Xác thực nội dung tập tin bị lỗi:**

Thay vì tin tưởng Content-Type trong yêu cầu, nhiều máy chủ bảo mật hơn sẽ cố gắng xác định nội dung của tập tin cho phù hợp.

Khi tải hình ảnh, máy chủ sẽ xác định thuộc tính của hình ảnh là kích thước. Nếu bạn tải tập lệnh PHP, nó sẽ không có kích thước. Máy chủ sẽ từ chối việc tải lên.

Tương tự, một số loại tập tin khác thường có chuỗi bytes riêng để đánh dấu. Ví dụ tập tin JPEG bắt đầu với FF D8 FF.

Đây là cách tốt hơn để xác thực loại tập tin. Nhưng với những công cụ đặc biệt như ExifTool có thể tạo ra tệp JPEG có chứa mã độc thì nó không còn đáng tin cậy nữa.

**Tấn công dồn dập để tải lên tệp:**

Những thư viện hiện đại đã kiên cường chống lại những loại tấn công này. Chúng không cho tải tập tin trực tiếp lên hệ thống. Thay vào đó, chúng cho tải tệp lên thư mục tạm thời, đầu tiên sẽ sàng lọc và đặt tên ngẫu nhiên để tránh ghi đè những tập tin đang có. Sau đó chúng xác thực trên tập tin tạm thời và chỉ chuyển nó đến hệ thống khi an toàn.

Nhưng những nhà phát triển thường xử lí tải tập tin độc lập với thư viện. Điều này rất nguy hiểm vì kẻ tấn công có thể tấn công dồn dập để bỏ qua lớp xác thực mạnh mẽ nhất.

Ví dụ, một vài trang web cho phép tải tập tin lên hệ thống nhưng xóa nó đi nếu không qua được xác thực. Cách làm này giống với phần mềm chống virus và kiểm tra mã độc. Nó chỉ mất vài mili giây nhưng trong khoảng thời gian ngắn ngủi mà tập tin tồn tại trong hệ thống, kẻ tấn công vẫn có thể kích hoạt được.

**Tấn công dồn dập khi đăng tải tập tin dựa trên Url:**

Bình thường khi đăng tải tập tin sẽ dùng phương thức POST. Tấn công dồn dập cũng áp dụng với đăng tải tập tin dựa trên Url. Trong trường hợp này , máy chủ sẽ lấy tập tin qua internet và sao chép nó trước khi thực hiện xác thực.

Khi tập tin được tải bằng http, những nhà phát triển không dùng thư viện để xác thực được. Thay vào đó , bọn họ phải tự tạo ra quá trình xử lí đăng tải tập tin 🡪 Thiếu bảo mật.

Ví dụ, nếu tập tin được tải trong thư mục tạm thời với tên ngẫu nhiên, kẻ tấn công không thể tấn công dồn dập. Nếu không biết tên thư mục, bọn họ sẽ không kích hoạt tập tin được. Mặc khác nếu tên thư mục ngẫu nhiên được tạo bằng hàm ngẫu nhiên giả như PHP’s uniqid(), nó vẫn brute-forced được.

Để tấn công dễ dàng hơn, bạn có thể cần kéo dài thời gian xử lí tập tin, do đó kéo dài thời gian để brute-forced tên thư mục. Bằng cách tải lên tập tin lớn

**Khai thác lỗ hổng tải tập tin không cần kích hoạt mã từ xa:**

**Việc tải lên những tập lệnh phía máy khách:**

Mặc dù bạn không thể kích hoạt tập lệnh trên máy chủ, bạn vẫn có thể tải tập lệnh để thực hiện tấn công phía máy khách. Ví dụ nếu bạn có thể tải html hay svg bạn có thể dùng thẻ <script> để tấn công Stored XSS.

Nếu tập tin được tải xuất hiện trên những trang được thăm bởi người dùng, trình duyệt sẽ kích hoạt tập lệnh khi nó trích xuất ra trang web.

Lưu ý là vì những hạn chế của chính sách chung nguồn gốc, những loại tấn công này chỉ hoạt động nếu tập tin tải lên được yêu cầu từ chung 1 nguồn gốc với trang web.

**Khai thác lỗ hổng trong phân tích cú pháp những tập tin đã tải lên:**

Nếu tập tin đã tải lên đều lưu trữ và cung cấp rất an toàn, cách cuối cùng là thử khai thác những lỗ hổng đặc thù là phân tích hay xử lí những định dạng tập tin khác nhau. Ví dụ, máy chủ sẽ phân tích những tập tin dựa trên XML như .doc hay .xls, đây có thể là 1 cuộc tấn công XXE.

**Việc tải tập tin bằng PUT:**

Cần lưu ý rằng một số máy chủ web có thể được cấu hình để hỗ trợ các yêu cầu PUT. Nếu không phòng thủ, đây có thể là cách khác để đăng tải những tập lệnh độc hại thậm chí ứng dụng không có chức năng đăng tải.

PUT /images/exploit.php HTTP/1.1

Host: vulnerable-website.com

Content-Type: application/x-httpd-php

Content-Length: 49

<?php echo file\_get\_contents('/path/to/file'); ?>

(Bạn có thể gửi những yêu cầu OPTIONS đến những vị trí khác nhau để kiểm tra xem máy chủ có hỗ trợ phương thức PUT không.)

**Làm sao để ngăn chặn lỗ hổng đăng tải tập tin:**

Việc cho phép đăng tải tập tin là bình thường và sẽ không nguy hiểm nếu bạn ngăn chặn đúng cách. Cách hiệu quả nhất để bảo vệ những trang web khỏi những lỗ hổng này là:

* Dùng whitelist để kiểm tra phần mở rộng tập tin hơn là blacklist.
* Đảm bảo rằng tên tập tin không chứa bất cứ chuỗi nào để truyền thư mục như ../
* Đổi tên những tập tin đã được tải lên để tránh bị ghi đè
* Không đăng tải những tập tin lên hệ thống file cho đến khi nó được xác thực đầy đủ
* Sử dụng 1 thư viện đã được thiết lập để xử lí trước việc đăng tải tập tin hơn là cố gắng viết cơ chế xác thực của mình.