**Rò rỉ thông tin**

**Khái niệm:**

Rò rỉ thông tin là khi 1 trang web không cố ý tiết lộ thông tin nhạy cảm cho người dùng. Những loại thông tin sau có thể bị kẻ tấn công khai thác như:

* Dữ liệu về người dùng như tài khoản hoặc thông tin tài chính
* Dữ liệu kinh doanh nhạy cảm
* Thông tin kĩ thuật về trang web và cơ sở hạ tầng của nó

Mặc dù một số thông tin sẽ được sử dụng hạn chế, nhưng nó có thể là khởi điểm để mở rộng phạm vi tấn công có thể chứa những lỗ hổng thú vị khác.

Rất hiếm khi, thông tin nhạy cảm bị rò rỉ cho người dùng đang duyệt trang web bình thường. Phổ biến hơn, kẻ tấn công cần khai thác thông tin bằng cách tương tác với trang web theo cách không mong muốn. Sau đó họ sẽ lấy được phản hồi của trang web để xác định những hành vi thú vị.

**Những ví dụ về rò rỉ thông tin:**

* Tiết lộ tên của các thư mục bị ẩn, cấu trúc và nội dung của nó qua tập tin robots.txt
* Được phép truy cập mã nguồn qua các bản sao lưu tạm thời
* Tên cơ sở dữ liệu, bảng và cột được nhắc đến trong thông báo lỗi
* Rò rỉ thông tin nhạy cảm như thẻ ngân hàng
* API keys để mã hóa, địa chỉ IP, thông tin đăng nhập của cơ sở dữ liệu trong mã nguồn
* Gợi ý về sự tồn tại hay thiếu các mã nguồn, tài khoản thông qua sự khác biệt nhỏ trong hành vi của ứng dụng.

**Lỗ hổng rò rỉ thông tin sinh ra như thế nào?**

Lỗ hổng rò rỉ thông tin có thể sinh ra bằng nhiều cách khác nhau, nhưng chúng có thể được phân loại như sau:

* **Failure to remove internal content from public content**. Ví dụ, người dùng có thể nhìn thấy comments của nhà phát triển.
* **Insecure configuration of the website and related technologies**. Ví dụ, không tắt được tính năng debug hay diagnostic có thể giúp kẻ tấn công nhận được thông tin nhạy cảm. Những cấu hình mặc định cũng có thể làm trang web bị dính lỗ hổng, ví dụ, hiển thị thông báo rõ ràng.
* **Flawed design and behavior of the application**. Ví dụ, nếu 1 trang web trả về các phản hồi khác nhau khi trạng thái lỗi khác nhau xảy ra, điều này cho phép kẻ tấn công liệt kê được dữ liệu nhạy cảm như thông tin đăng nhập của người dùng.

**Ảnh hưởng của lỗ hổng rò rỉ thông tin**

Lỗ hổng rò rỉ thông tin có thể có ảnh hưởng trực tiếp hay gián tiếp dựa vào mục đích của trang web và do đó, kẻ tấn công có thể lấy được thông tin gì. Trong một số trường hợp, việc tiết lộ thông tin nhạy cảm có thể ảnh hưởng nghiêm trọng đến bên thứ ba. Ví dụ, cửa hàng online để lộ thông tin thẻ ngân hàng của khách hàng.

Mặc khác, rò rỉ thông tin kĩ thuật như cấu trúc thư mục hay thư viện của bên thứ ba đang dùng có thể ít tác động trực tiếp. Tuy nhiên, nó có thể là thông tin quan trọng để thực hiện nhiều khai thác khác. Mức độ nghiêm trọng sẽ dựa vào kẻ tấn công có thể làm gì với thông tin đó.

**Cách đánh giá mức độ nghiêm trọng của việc rò rỉ thông tin**

Trong quá trình kiểm thử, rò rỉ thông tin kĩ thuật chỉ thường nguy hiểm nếu bạn có thể mô phỏng được cách kẻ tấn công có thể gây hại với nó.

Ví dụ, 1 trang web đang dùng 1 thư viện có phiên bản mới nhất đã được vá đầy đủ thì sẽ không gây hại gì. Nhưng thông tin này sẽ rất quan trọng khi trang web dùng 1 phiên bản cũ hơn chứa lỗ hổng có sẵn. Trong trường hợp đó, thực hiện 1 cuộc tấn công có thể đơn giản như public exploit.

Điều quan trọng là phải thử nhiều cách khai thác khác nhau khi bạn tìm được thông tin nhạy cảm. Những thông tin kĩ thuật nhỏ có thể được khám phá bằng nhiều cách khác nhau ở trên trang web. Do đó, trọng tâm chính của bạn phải là tác động và khả năng khai thác của thông tin bị rò rỉ.

**Tìm kiếm và khai thác lỗ hổng rò rỉ thông tin**

**Làm sao để kiểm tra lỗ hổng rò rỉ thông tin?**

Bạn nên tránh tập trung vào 1 lỗ hổng cụ thể. Dữ liệu nhạy cảm có thể bị rò rỉ ở khắp nơi.

Những kĩ thuật cao và công cụ bạn có thể dùng để xác định các lỗ hổng rò rỉ dữ liệu.

**Fuzzing**

Nếu bạn xác định được những tham số thú vị, bạn có thể gửi các loại dữ liệu mà ứng dụng không mong muốn để xem nó phản hồi như thế nào. Mặc dù những phản hồi thường tiết lộ nhiều thông tin thú vị, nhưng chúng cũng có thể ẩn chứa trong hành vi của ứng dụng. Ví dụ, nó có thể khác nhau về thời gian ứng dụng xử lí yêu cầu. Thậm chí nếu thông báo lỗi không tiết lộ bất cứ thứ gì, nhưng 2 lỗi khác nhau cũng là 1 thông tin hữu ích.

Bạn có thể tự đọng quá trình này bằng công cụ như Burp Intruder. Nó cung cấp rất nhiều lợi ích. Đáng chú ý nhất, bạn có thể:

* Thêm vị trí của payload đến các tham số và dùng 1 wordlist có sẵn để kiểm tra liên tiếp 1 lượng lớn các đầu vào khác nhau.
* Dễ dàng xác định sự khác nhau trong các phản hồi bằng cách so sánh trạng thái code http, thời gian phản hồi,…
* Dùng grep matching rules để nhanh chóng xác định các keywords như error, invalid, SELECT, SQL,…
* Dùng grep extraction rules để trích xuất và so sánh nội dung của các phản hồi.

Bạn có thể dùng Logger++, có sẵn trong Bapp store. Ngoài việc ghi lại yêu cầu và phản hồi từ các công cụ của Burp, nó cho phép bạn sàng lọc những thông tin quan trọng của ứng dụng. Đây chỉ là 1 trong những extensions của Burp có thể giúp bạn tìm kiếm thông tin nhạy cảm được rò rỉ bởi trang web.

**Using Burp Scanner**

**Using Burp's engagement tools**

Burp cung cấp 1 số công cụ tương tác cho phép bạn tìm kiểm thông tin thú vị trong trang web.

* **Search:** bạn có thể dùng công cụ này để tìm kiếm bất cứ biểu thức nào. Bạn có thể điều chỉnh kết quả bằng các lựa chọn tìm kiếm nâng cao như regex hay negative. Điều này rất hữu ích để tìm kiếm các keyword bạn thích.
* **Find comments:** trích xuất các comments của nhà phát triển
* **Discovery content:** xác định các nội dung và chức năng mà trang web không hiển thị. Điều này rất hữu ích để tìm thêm các thư mục và tập tin bị ẩn.

**Engineering informative responses:**

Các thông báo lỗi rõ ràng có thể tiết lộ nhiều thông tin thú vị trong quá trình bạn kiểm thử. Tuy nhiên, nghiên cứu sự thay đổi của các thông báo lỗi giúp bạn tiến xa hơn. Trong một số trường hợp, bạn có thể tận dụng trang web để trích xuất dữ liệu qua thông báo lỗi.

Một cách phổ biến nhất là làm cho logic của ứng dụng thực hiện 1 hành động không mong muốn trên 1 mục dữ liệu cụ thể. Ví dụ, gửi 1 giá trị của tham số không hợp lệ có thể dẫn đến stack trace hay debug chứa thông tin quan trọng. Đôi khi, bạn có thể làm cho thông báo lỗi hiển thị dữ liệu mình mong muốn.

**Những nguồn rò rỉ thông tin phổ biến:**

Việc rò rỉ thông tin có thể xảy ra ở rất nhiều bối cảnh trong 1 trang web.

**Files for web crawlers**

Rất nhiều trang web cung cấp những tập tin ở /robots.txt và /sitemap.xml để giúp crawlers chuyển hướng trang dễ hơn. Những tập tin này thường liệt kê các thư mục mà crawlers nên bỏ qua vì chúng có thể chứa dữ liệu nhạy cảm.

Vì những tập tin này không được liên kết với trang web nên chúng không thể xuất hiện trong Burp site map. Tuy nhiên, rất đáng để chuyển hướng đến /robots.txt hay /sitemap.xml để xem có cái gì hữu ích không.

**Directory listings**

Những máy chủ web có thể được cấu hình để tự động liệt kê nội dung của các thư mục không có trang index. Điều này có thể giúp kẻ tấn công nhanh chóng xác định mã nguồn tại 1 đường dẫn nhất định, phân tích và tấn công chúng.

Liệt kê thư mục không phải là 1 lỗ hổng nghiêm trọng. Tuy nhiên, nếu trang web thất bại trong kiểm soát truy cập, tiết lộ vị trí của những mã nguồn nhạy cảm thì mới là vấn đề nghiêm trọng.

**Developer comments**

Trong quá trình phát triển phần mềm, những comment html được thêm để đánh dấu. Khi ra sản phẩm, chúng thường bị loại bỏ. Tuy nhiên, chúng có thể bị quên hay cố ý để lại vì một vài người không quan tâm đến bảo mật. Mặc dù những comments không nhìn được trên trang web nhưng chúng có thể dễ dàng truy cập qua Burp hay developer tool.

Đôi khi comment chứa thông tin hữu ích cho kẻ tấn công như gợi ích về sự tồn tại của 1 thư mục bị ẩn hay manh mối về logic của ứng dụng.

**Error messages**

Một trong những nguyên nhân chính gây ra rò rỉ thông tin là thông báo lỗi quá rõ ràng. Là 1 nguyên tắc chung, bạn nên chú ý tất cả thông báo lỗi khi kiểm tra.

Thông báo lỗi có thể tiết lộ thông tin về loại dữ liệu hay đầu vào mong muốn của 1 tham số xác định. Nó có thể giúp bạn thu hẹp phạm vi tấn công bằng cách xác định tham số có thể khai thác. Nó có thể ngăn chặn bạn mất thời gian để cố gắng tiêm payload sẽ không hoạt động

Thông báo lỗi rõ ràng có thể cung cấp thêm thông tin về công nghệ đang được dùng bởi ứng dụng. Ví dụ, tên của 1 thư viên, loại cơ sở dữ liệu hay máy chủ trang web đang dùng cùng với phiên bản của nó. Thông tin này rất có ích vì bạn có thể tìm kiếm những khai thác có thể tồn tại ở phiên bản đó. Tương tự bạn có thể kiểm tra xem có các lỗi về cấu hình hay cài đặt mặc định gây nguy hiểm mà bạn có thể khai thác.

Bạn cũng có thể khám phá trang web đang dùng thư viện mã nguồn mở nào. Trong trường hợp đó, bạn có thể học mã nguồn công khai là 1 mã nguồn vô giá để thực hiện khai thác của riêng bạn.

Sự khác nhau giữa các thông báo lỗi cũng có thể là hành vi khác nhau của ứng dụng mà nhờ vào đó, chúng ta có thể tấn công SQL Injection, liệt kê tên người dùng ….

**Debugging data**

Để sửa lỗi, rất nhiều trang web tạo ra thông báo lỗi tùy chỉnh và logs chứa lượng lớn thông tin về hành vi của ứng dụng. Trong khi thông tin này rất hữu ích với lập trình viên, nó cũng rất hữu ích với kẻ tấn công nếu bị rò rỉ.

Những thông báo sửa lỗi có thể chứa thông tin quan trọng để thực hiện 1 cuộc tấn công bao gồm:

* Tên máy chủ và thông tin đăng nhập trên hệ thống
* Tên tập tin và thư mục trên máy chủ
* Các keys để mã hóa dữ liệu truyền qua máy khách
* Giá trị của biến session có thể bị lạm dụng qua đầu vào người dùng

Thông tin sửa lỗi có thể được ghi lại trong 1 file riêng. Nếu kẻ tấn công có thể giành quyền truy cập file này, nó sẽ là 1 tham khảo hữu ích để hiểu về cách vận hành của ứng dụng. Nó cũng cung cấp nhiều manh mối về cách họ có thể cung cấp đầu vào để thao túng trạng thái ứng dụng và kiểm soát thông tin nhận được.

**User account pages**

Hồ sơ của người dùng hay trang tài khoản thường chứa thông tin nhạy cảm như địa chỉ email, số đt, API key,… Người dùng bình thường chỉ có thể truy cập trang tài khoản của chính họ. Tuy nhiên, một vài trang web chứa lỗi logic cho phép kẻ tấn công tận dụng những trang đó để xem dữ liệu của người dùng khác.

Ví dụ, 1 trang web tải trang tài khoản của người dùng dựa vào tham số user:

GET /user/personal-info?user=carlos

Đa số trang web sẽ ngăn chặn kẻ tấn công thay đổi tham số đó để truy cập tài khoản bất kì. Nhưng vẫn có lỗi logic để tải các mục dữ liệu nhỏ.

Kẻ tấn công không thể tải toàn bộ trang tài khoản của người dùng khác, nhưng logic để fetch và kết xuất địa chỉ email thì được. Ví dụ, không thể kiểm tra tham số user có đúng với người dùng đang đăng nhập không. Trong trường hợp đó, thay đổi tham số user cho phép kẻ tấn công hiển thị địa chỉ email của người dùng bất kỳ.

**Backup files**

Truy cập được mã nguồn giúp kẻ tấn công hiểu được hành vi của ứng dụng và xây dựng các cuộc tấn công có mức độ nghiêm trọng cao. Dữ liệu nhạy cảm đôi khi còn được mã hóa trong mã nguồn.

Nếu bạn có thể xác định công nghệ có mã nguồn mở đang được dùng, điều này có thể giúp truy cập mã nguồn dễ dàng.

Rất hiếm khi, một trang web hiển thị mã nguồn. Khi ánh xạ 1 trang web, bạn có thể tìm thấy mã nguồn được tham chiếu rõ ràng. Không may mắn, nó không tiết lộ code được. Khi một máy chủ xử lí tập tin với extension cụ thể như php, nó sẽ thực thi code hơn là gửi đến máy khách như 1 văn bản. Tuy nhiên, trong một số tình huống, bạn có thể lừa trang web trả về nội dung của tệp. Ví dụ, text editor thường tạo 1 tệp sao lưu khi bản gốc được chỉnh sửa. Những tệp tạm thời này có thể được chỉ định bằng cách thêm (~) vào filename hay thêm 1 extension khác. Việc yêu cầu 1 bản sao lưu của mã nguồn cho phép bạn đọc được nội dung của nó trong phản hồi.

Khi kẻ tấn công truy cập được mã nguồn, đây là 1 bước quan trọng để xác định và khai thác những lỗ hổng khác như insecure deserialization.

**Insecure configuration**

Các trang web thường bị dính lỗ hổng vì cấu hình sai. Điều này thường phổ biến do việc sử dụng rộng rãi các công nghệ của bên thứ ba, với vô số tùy chọn cấu hình mà những người triển khai chúng không nhất thiết phải hiểu rõ.

Trong nhiều trường hợp khác, nhà phát triển có thể quên tắt lựa chọn debugging. Ví dụ, phương thức http TRACE được thiết kế nhằm mục đích phân tích. Máy chủ web sẽ phản hồi các yêu cầu dùng phương thức TRACE bằng cách lặp lại trong phản hồi chính xác yêu cầu đã nhận được. Hành vi này trông có vẻ vô hại, nhưng đôi khi dẫn đến rò rỉ dữ liệu như tên của các headers xác thực nội bộ có thể được thêm vào các yêu cầu bằng reverse proxies.

**Version control history**

Tất cả trang web ảo được phát triển bằng một số hệ thống kiểm soát phiên bản như Git. Mặc định, Git project lưu trữ tất cả dữ liệu kiểm soát phiên bản trong thư mục .git . Rất hiếm khi, các trang web hiển thị thư mục đó trong môi trường. Trong trường hợp này, bạn có thể truy cập nó bằng cách duyệt qua /.git.

Trong khi duyệt thủ công qua cấu trúc tệp và nội dung là phi thực tế, có rất nhiều cách khác nhau để tải toàn bộ thư mục .git. Sau đó bạn có thể mở nó bằng ứng dụng Git để giành quyền truy cập version control history của trang web. Nó có thể bao gồm logs chứa những lần commits và thông tin thú vị khác.

Nó không giúp bạn truy cập được mã nguồn nhưng so sánh sự khác nhau sẽ giúp bạn đọc 1 đoạn code nhỏ. Giống như bất kì mã nguồn nào, bạn cũng có thể thấy dữ liệu nhạy cảm trong một số dòng đã thay đổi.