SECURITYBOAT
Frontline Of Your Business

UNRESTRICTED FILE UPLOAD
HANDBOOK

www.securityboat.net

SECURITYBOAT
Frontline Of Your Business

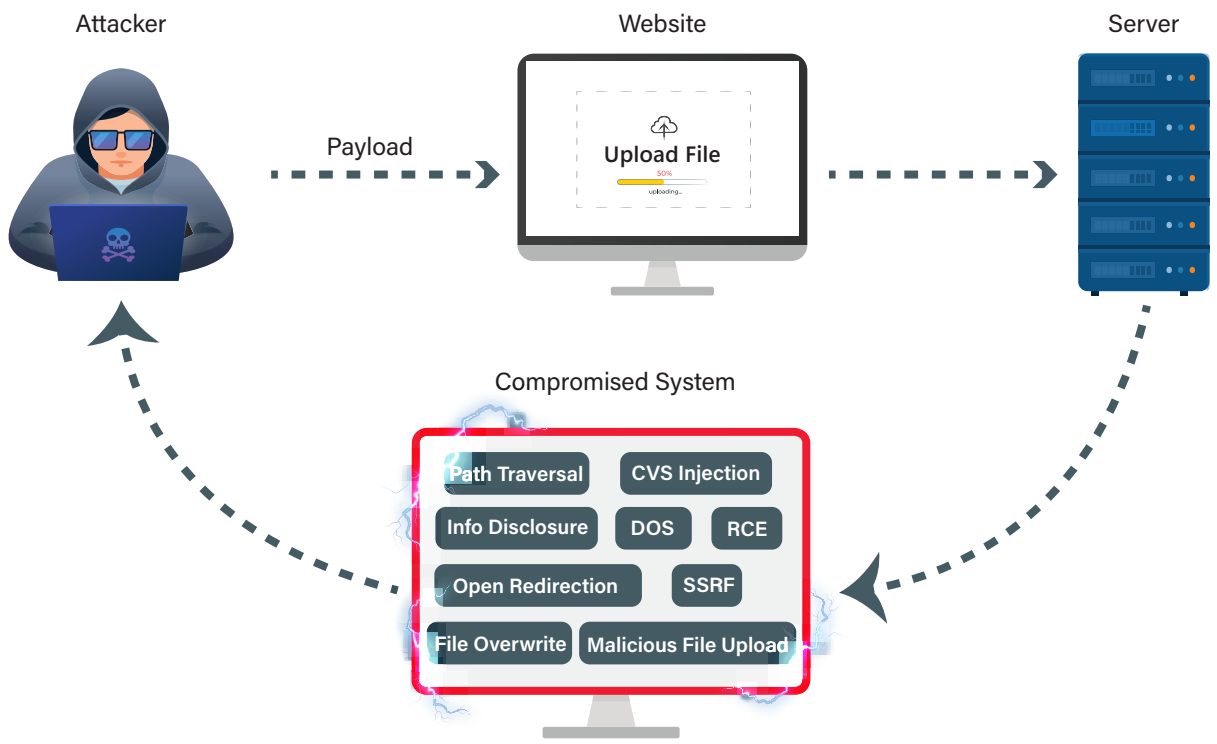
Table of contents

1. What is Unrestricted File Upload Vulnerability?.....01
2. How do vulnerabilities related to file uploads emerge?..... 01
3. File Upload Attack.....02
   3.1 RCE.....02
   3.2 Pixel Flood Attack.....02
   3.3 CSV Injection.....03
   3.4 XSS.....03
4. Bypasses.....04
   4.1 Bypass File Extensions checks.....04
   4.2 Bypass Content-Type checks.....04
   4.3 Bypass Blacklisted Extension.....04
5. Impact.....05
   5.1 RCE.....05
   5.2 Overwriting Critical Files.....05
   5.3 DOS.....05
   5.4 Web Defacement.....05
   5.5 Phishing Page.....05
6. Prevention.....06
7. Additional Resources and References.....07
   7.1 Unrestricted File Upload Vulnerability..... 07
   7.2 File Upload Attack Techniques.....07
   7.3 File Upload Restrictions Bypass.....07
   7.4 File Upload Attack Prevention..... 07
8. QR Code.....08

SECURITYBOAT
Frontline Of Your Business

1. What is Unrestricted File Upload Vulnerability?

File upload vulnerabilities in web applications occur when the application does not properly validate or restrict the types of files that users can upload. Attackers can exploit this weakness by uploading malicious files, such as scripts or executable files, which can then be executed on the server. This can lead to various security issues, including remote code execution, denial of service, or unauthorized access.



Common File Upload Attack Flow

2. How do File Uploads Vulnerabilities emerge?

Despite the evident risks, it's uncommon to find websites in the wild without any restrictions on the types of files users can upload. Typically, developers implement what they believe to be strong validation mechanisms, which may either have inherent flaws or can be easily circumvented.

For instance, developers might try to create a blacklist of hazardous file types but overlook parsing discrepancies when checking file extensions. Like any blacklist, there's also the risk of unintentionally excluding more obscure file types that could still pose a threat. In other scenarios, websites may attempt to verify the file type by examining properties that an attacker can easily manipulate using tools like Burp Proxy or Repeater.

In the wild, file upload vulnerabilities can stem from inadequate file type validation, absence of size limits, weak input validation, improper file permissions, failure to scan for malware, and insufficient authentication or authorization checks. Attackers may exploit these weaknesses to upload malicious files, execute unauthorized actions, or compromise the system's integrity.

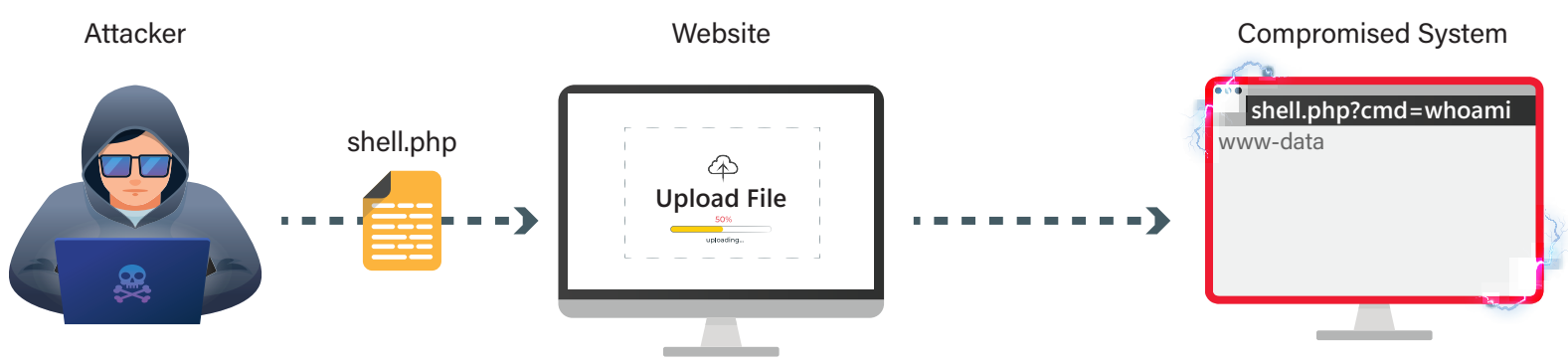
Ultimately, even when robust validation measures are in place, there may be inconsistencies in their application across the network of hosts and directories that constitute the website. These disparities create openings that can be exploited by attackers.

3. File Upload Attack

3.1 RCE

Remote Code Execution (RCE) through file upload is a critical security vulnerability that occurs when a website or application allows users to upload files, and these files are not properly validated or sanitized. In the worst-case scenario, the system may permit the upload and execution of server-side scripts, such as those written in PHP, Java, or Python.

To attain remote code execution, the process involves utilizing a PHP, JSP (or any other) shell, uploading it while circumventing any restrictions, checking accessibility at the specified path, and subsequently attempting shell execution based on the method of shell execution (web shell, command-line shell).

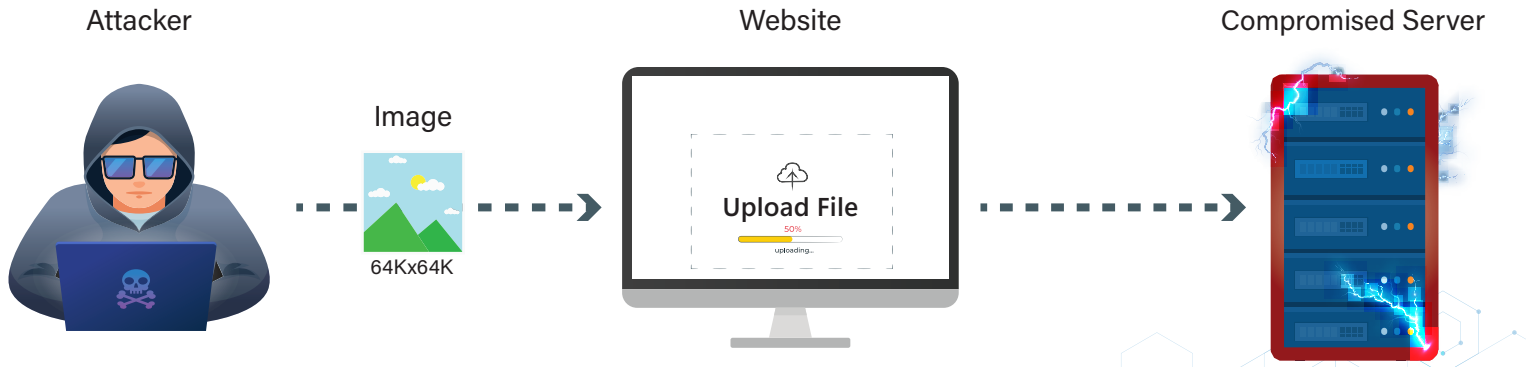


Remote Code Execution

3.2 Pixel Flood Attack

A straightforward assault that can be evaluated whenever a file upload feature is enabled for image files. In a Pixel Flood Attack, the assailant endeavours to upload a file with an excessively large pixel size, causing the server to exhaust its resources and potentially leading to application crashes. This simplistic method can result in a denial-of-service scenario at the application level. Many contemporary applications rely on third-party libraries for processing images, converting them into smaller sizes to conserve storage and processing resources.

To execute a Pixel Flood Attack, create an image with dimensions of 64,250 x 64,250 pixels, then navigate to the susceptible application offering image file uploads. Upload the oversized image and monitor the server's response. If there's a significant delay or if the application becomes inaccessible, verify the issue on another device. If the lag or accessibility problem persists, it confirms the vulnerability of the application to a Pixel Flood Attack.

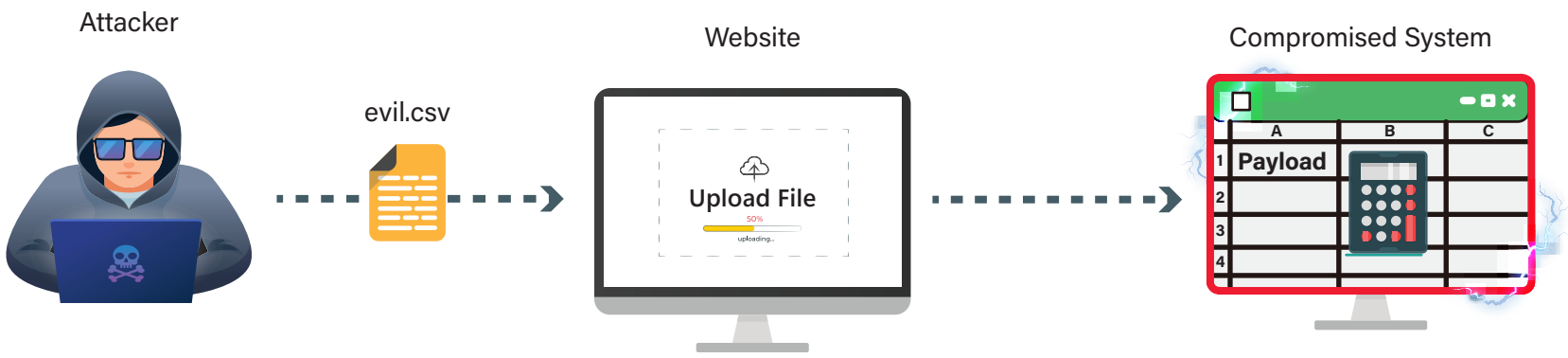


Pixel Flood Attack

3.3 CSV Injection

CSV Injection, also known as Formulae Injection or CSV Excel Macro injection, is a vulnerability typically observed in the "File Export" feature rather than the "File Upload" functionality. Although there are instances where an application diligently sanitizes user-provided input, preventing the inclusion of malicious payloads even with client-side validation bypass, this effectively thwarts potential attacks.

To identify CSV Injection, upload a CSV file embedded with a malicious payload. Next, export the uploaded content using a different user account within the application. If the application neglects to properly sanitize the user-provided content during file output, it may leave room for the successful execution of the injection attack.

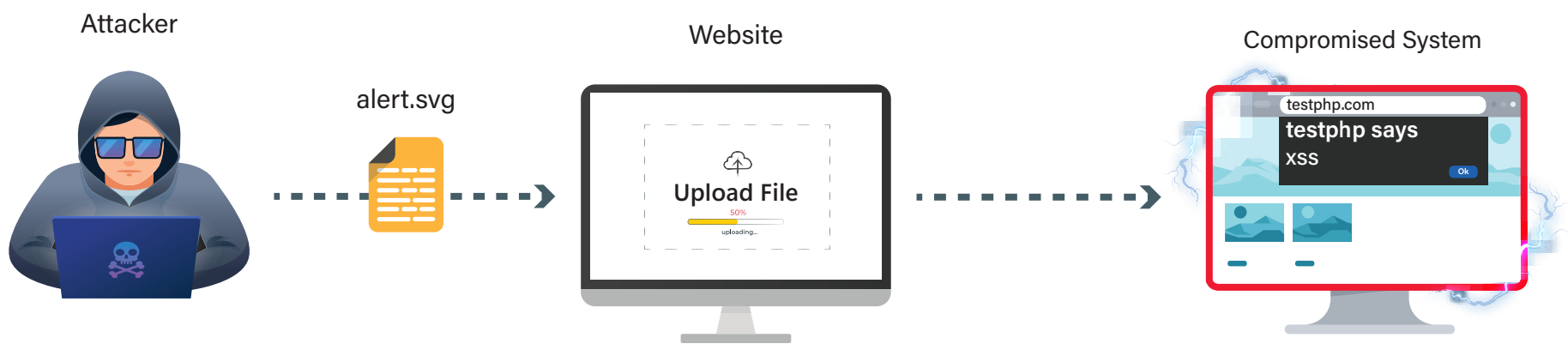


CSV Injection Attack

3.4 XSS

During the examination of file upload functionality, various methods can be employed to carry out a cross-site scripting attack. One approach involves uploading malicious files, such as SVG or HTML files, and altering the file name to incorporate a cross-site scripting payload.

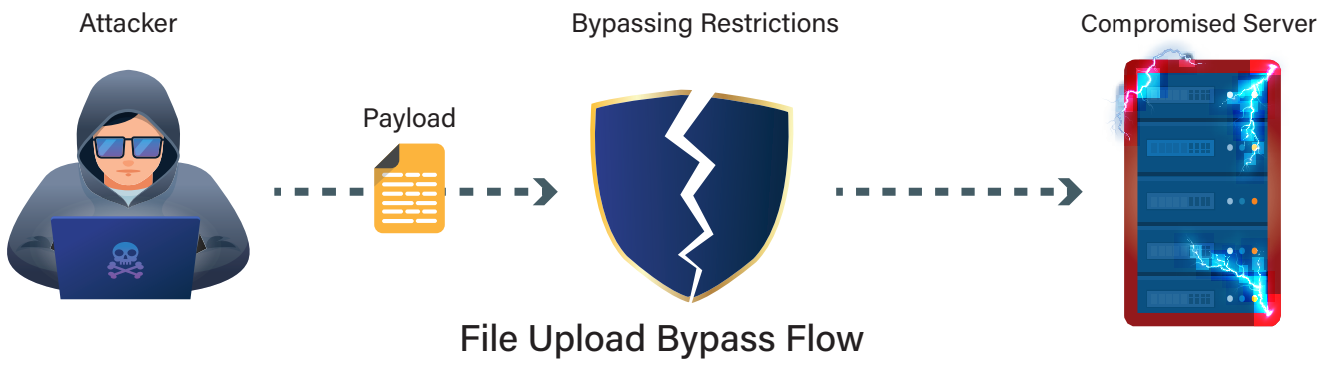
To leverage XSS using an SVG File via Unrestricted File Upload Vulnerability, one should generate an SVG file embedded with a cross-site scripting payload. Subsequently, access the file upload feature and upload the crafted SVG file. Either open the SVG file directly or visit the endpoint invoking the SVG file; if the application possesses the vulnerability, a Cross-Site Scripting execution will be evident.



Cross Site Scripting Attack

4. Bypasses

While there is a broad range of file upload-related attacks, modern applications concurrently incorporate varying levels of protection to mitigate or, at the very least, minimize the risk and probability of exploitation.



File Upload Bypass Flow

Nonetheless, there exist several methods to circumvent restrictions associated with file upload attacks. The following are some noteworthy bypass techniques:

4.1 Bypass File Extensions Checks

If the client-side validation is designed to check the filename attribute of the uploaded file as a restriction, then an attacker can easily bypass it by manipulating the file extension in filename.

```
Modified Request
----------------
POST /upload
Host: vulnerable_target.com
....
other headers
....content_type=image/png&filename=test.php.jpg&data={file_content}
```

Modify request to Bypass File Extensions Check

4.2 Bypass Content-Type Checks

It is possible to change the content type to mimic the malicious file as one of the allowed file extension types, resulting in bypassing the restrictions and uploading the malicious files.

```
Modified Request
----------------
POST /upload
Host: vulnerable_target.com
....
other headers
....content_type=image/png&filename=test.php&data={file_content}
```

Modify request to Bypass Content-Type Check

4.3 Bypass Blacklisted Extension

If an application is using a blacklist of extension, it is possible to bypass the restriction by trying for similar extensions with different capitalization or versions. Example:

Blocked File: test.php
Allowed File: test.php3, test.PhP, test.PHp

5. Impact

As with many other vulnerability classes, there is no single answer to what file upload vulnerabilities can do to a target system. It heavily depends on the web application code written by developers, on the web server configuration, as well on the operating system running the web server. The impact of file upload vulnerabilities generally depends on few factors:

Factor one: Which aspect of the file the website fails to validate properly, whether that be its size, type, contents, and so on.

Factor two: What restrictions are imposed on the file once it has been successfully uploaded. We will go ahead and have a look at some typical Impact scenarios:

5.1 RCE

In the worst-case scenario, the file's type isn't validated properly, and the server configuration allows certain types of files (such as .php and .jsp) to be executed as code. In this case, an attacker could potentially upload a server-side code file that functions as a web shell, effectively granting them full control over the server.

5.2 Overwriting Critical Files

If the filename isn't validated properly, this could allow an attacker to overwrite critical files simply by uploading a file with the same name. If the server is also vulnerable to directory traversal, this could mean attackers are even able to upload files to unanticipated locations.

5.3 DOS

Failing to make sure that the size of the file falls within expected thresholds could also enable a form of denial-of-service (DoS) attack, whereby the attacker fills the available disk space, overloading the server to work on client requests.

5.4 Web Defacement

If the web root is not configured properly (allowing an attacker to overwrite existing files), an attacker could substitute existing web pages with his own content (potentially showing imagery which is conflicting to the original purpose of the application)

5.5 Phishing Page

Like the example before, an attacker could also go ahead only slightly manipulate an existing page to e.g. extract sensitive data, sending it to a destination controlled by himself.

6. Prevention

Implementing File Upload Security Measures:

File Type Restriction: Restrict the allowed file types to prevent the upload of executables, scripts, and potentially harmful content.

File Type Verification: Confirm that files are not masquerading as allowed types. Verify file types independently of file extensions to avoid potential security bypasses.

Malware Scanning: Scan all uploaded files for malware using multiple anti-malware engines, employing signatures, heuristics, and machine learning detection methods for comprehensive threat detection.

Embedded Threat Removal: Employ Content Disarm and Reconstruction (CDR) to eliminate possible embedded threats in files like Microsoft Office, PDFs, and images, which may not be detected by standard anti-malware engines.

User Authentication: Enhance security by requiring user authentication before allowing file uploads, although this does not guarantee the user's machine integrity.

Size and Name Restrictions: Set maximum name length and file size limits, restricting allowed characters in names, if possible, to prevent potential service disruptions.

Randomized File Names: Randomly alter uploaded file names to thwart attackers attempting to access files using the originally uploaded name, especially when utilizing Content Disarm and Reconstruction (CDR).

Secure File Storage: Store uploaded files outside the web root folder to prevent attackers from executing files through assigned path URLs.

Vulnerability Checks: Before uploading, examine software and firmware files for vulnerabilities to prevent potential security risks.

Simple Error Messages: Display concise error messages without revealing directory paths, server configurations, or other sensitive information that could be exploited by attackers for unauthorized access.

7. Additional Resources and References

7.1 Unrestricted File Upload Vulnerability

- https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload
- https://portswigger.net/web-security/file-upload

7.2 File Upload Attack Techniques

- https://www.yeswehack.com/learn-bug-bounty/file-upload-attacks-part-1
- https://www.yeswehack.com/learn-bug-bounty/file-upload-attacks-part-2
- https://book.hacktricks.xyz/pentesting-web/file-upload

7.3 File Upload Restrictions Bypass

- https://workbook.securityboat.net/resources/web-app-pentest/unristricted-file-upload

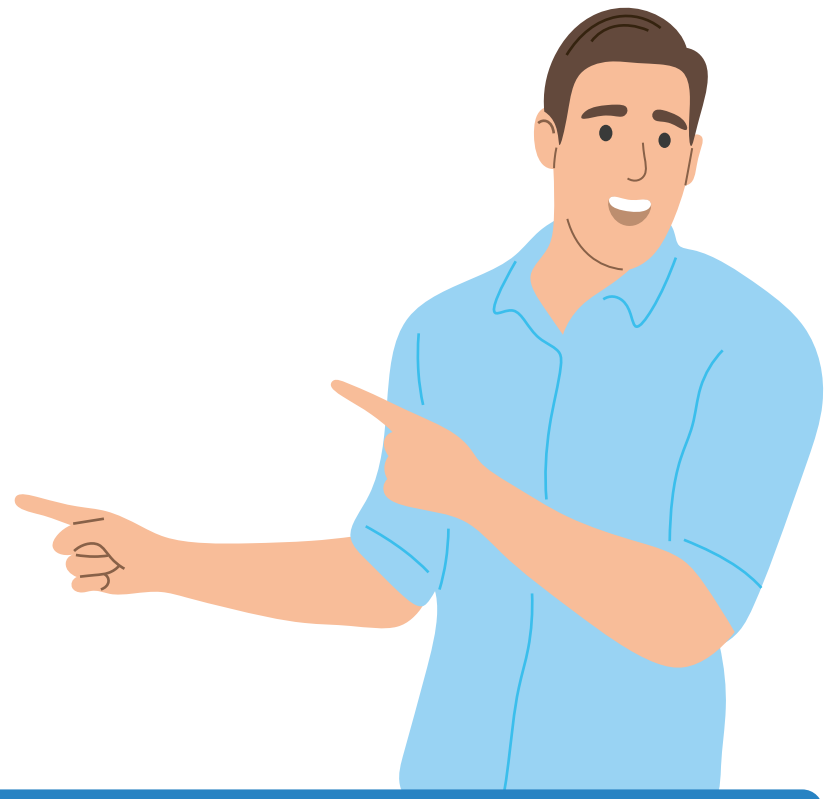
7.4 File Upload Attack Prevention

- https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html

SECURITYBOAT

Frontline Of Your Business

UNRESTRICTED FILE UPLOAD HANDBOOK



Scan QR Code to Download Handbook

www.securityboat.net