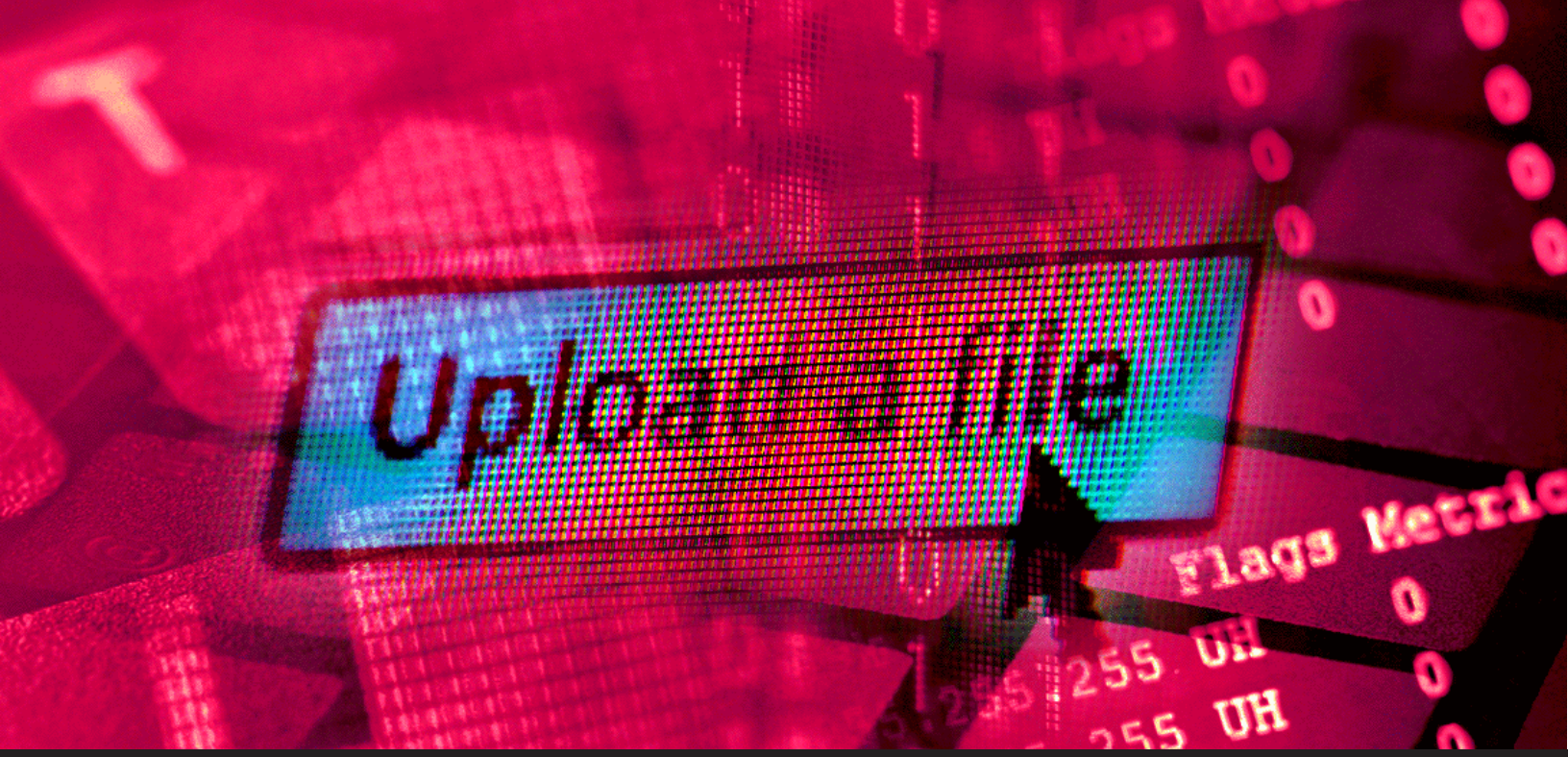

How to Prevent File Upload Vulnerabilities

Devolutions

USER-GENERATED FILE UPLOADS ARE ESSENTIAL FOR MANY APPLICATIONS AND BUSINESS SERVICES

User-generated file uploads are essential for many applications and business services. For example, file uploads are a fundamental function for healthcare portals, Content Management Systems (CMS), and messaging applications.

However, allowing users to upload files comes with its own set of risks. Attackers are constantly trying to breach systems and steal information by embedding malicious content. Fortunately, you can avoid this type of crisis with proper preventative techniques. This article covers automatic file upload risks and types, as well as eight tips to prevent attacks.

File Upload Security Risks

Attackers can exploit non-secure file upload techniques to upload malicious content to systems. Here are the three most common risks of user-generated file uploads.

Unrestricted file types

Anyone who has access to your website can upload a malicious file to the server if you do not restrict the upload of certain file types, including Windows files like .exe, .pif, .bat. These file types are dangerous because they are capable of executing commands and running malicious codes.

Multipurpose Internet Mail Extensions (MIME) type validation

Attackers can easily bypass MIME type validation security to examine the content of a particular file. For example, MIME sniffing is a legitimate technique to determine the format of a file. However, attackers can leverage MIME sniffing to execute Cross Site Scripting (XSS) attacks.

Blacklisting file extensions

Blacklisting file extensions keeps track of potentially harmful extensions. When a user uploads a file, the system checks the file extension to make sure it is not on the blacklist. If it is, the file is rejected. Unfortunately, this method may not be able to list all harmful extensions. An attacker can use an extension that is not included on the list to deceive the security system.

Types of File Upload Attacks

There are 4 file upload vulnerability groups. When developing a system that accepts user-generated files, you should evaluate the risks related to each group. Also, you have to implement appropriate security checks to prevent attacks.

File size vulnerabilities

Unusually large files can lead to an overload or failure in an application. For instance, attackers can execute botnet or Denial of Service (DDoS) attacks to trigger simultaneous uploads of very large files. As a result, the system does not fulfill legitimate requests and eventually shuts down.

Malicious content

Uploaded file content can include exploits, malware and malicious scripts. An attacker can use malicious

content to manipulate the application behavior. For example, hackers can reveal a system access key by uploading specific malware.

File access vulnerabilities

Attackers can manipulate the access rules of files to penetrate vulnerable systems. For instance, hackers can get access to private user photos by misconfiguring AWS S3 configurations.

File metadata vulnerabilities

An incorrect file name or path can trick an application into copying the file to another location. This can result in file changes and lead to unexpected behavior. For instance, an attacker can overwrite important configuration files by using control characters in the file name. Another example is changing security settings to upload malicious files.

How to Prevent File Upload Vulnerabilities: 7 Best Practices

Follow these best practices to prevent the file upload attacks mentioned above:

1. File type verification

File types are usually defined by their file extensions. Each file type usually has several corresponding file extensions. The file extensions enable the operating system and users to easily identify the type of file.

Attackers can bypass security systems and spoof operating systems and users by changing file extensions. For example, hackers can rename a malicious .exe file into a legitimate-looking .docx file. To prevent this, you must verify the file type before allowing upload.

2. Restrict specific file extensions

A whitelist provides system access only to administrator-approved programs, IPs and email addresses. Creating a white list of allowed files enables you to avoid uploads of potentially malicious content to your site. The white list can include executables, scripts and any other file type.

3. Malware prevention

Websites that insert or parse data from uploaded files may be vulnerable to malware attacks. To prevent malware, you should scan all uploaded files with multiple anti-malware tools. Each tool uses different algorithms and specializes in different categories.

4. Remove embedded threats

Anti-malware tools don't always detect embedded threats in PDFs, MS Office and image files. For example, attackers use digital watermarking techniques to embed malicious code inside an image or video file. Make sure to remove any possible embedded objects from your uploaded files.

5. User authentication

User authentication methods validate the identity of the person requesting private information. Implement robust user authentication protocols like Two-factor authentication (2FA).

Two-factor authentication is a two-step authentication process. The process combines a password and username with a physical or mobile token for extra security. The sequence of authentication factors makes it more difficult for a potential intruder to gain access.

6. Store files in an external directory

Upload files to external directories and store them outside the webroot. This technique prevents attackers from executing malicious files through a website URL.

7. Simple error messages

Error messages sometimes use server configuration settings or directory paths to give more information to the user. However, cybercriminals can use the information from error messages to exploit vulnerabilities in uploaded files. For this reason, you should make the error message as simple as possible.

Advanced Protection Techniques for Enterprise-Grade Applications

The tips above cover the fundamentals of secure file uploads. However, you must consider additional protection levels when developing applications that require robust security. Here are some enterprise-grade file upload security techniques you can use:

- Protect your files from Cross-site Scripting (XSS) attacks — Attackers use XSS attacks to inject malicious code into legitimate web applications or files.
- Avoid using the HTTP PUT Method — The PUT method is designed to manage file operations. Attackers can use this method to upload malicious resources like web shells to a server. Instead, select encoded methods like POST.

- Audit write access to important configuration files — Use “web.config” or .htaccess” to block access to the file uploading system. You can do this manually or via an automatic file uploads system.
- Restrict access to certain web services without validating client caches — This is done by disabling your browser caching for corssdomain.xml and clientaccesspolicy.xml.

Conclusion

Automatic file uploads help organizations keep up with the extensive amount of user-generated data on the web. However, developing a secure file upload system is challenging. You must invest in file upload security to prevent costly data breaches that can have a significant impact on your organization. You can use the best practices above to identify cost-effective ways to manage and evaluate automatic file upload security.