

**STEFLAN**[\\_\(https://steflan-security.com\)](https://steflan-security.com)

# File Upload Restriction Bypass



## Checklist

CHECKLISTS ([HTTPS://STEFLAN-SECURITY.COM/CATEGORY/RESOURCES/CHECKLISTS/](https://steflan-security.com/category/resources/checklists/)),

RESOURCES ([HTTPS://STEFLAN-SECURITY.COM/CATEGORY/RESOURCES/](https://steflan-security.com/category/resources/))

# File Upload Restriction Bypass Checklist

January 20, 2021 | by Stefano Lanaro | [Leave a comment \(https://steflan-security.com/file-upload-restriction-bypass-cheat-sheet/#respond\)](https://steflan-security.com/file-upload-restriction-bypass-cheat-sheet/#respond)


## Introduction

When enumerating web applications, we often find ourselves in front of a file upload file that allows us to potentially upload malicious files onto the application, such as a PHP or ASP shell, although these will often have certain restrictions that will only allow certain file types, extensions, file names or contents.

Through this checklist, I hope to cover most of the possible bypass methods that can be used to get past this restriction.

## Checklist

Method	Details

Try various file extensions	Try different versions of the file extensions, for example php3, .php4, .php5, phtml for PHP scripts, asp,aspx and ashx for IIS 
Append an extra file extension	If the application is not properly validating for the file extension, this can be exploited by appending another extension, for example from script.php to script.php.gif or script.gif.php
Change the casing of the extension	Try different combinations of lower and upper case, for example pHp, PhP, phP, Php etc
Change content type	When intercepting the request using Burp Suite, the content type can be changed, for example from "Content-type: application/x-php" to "Content-type: image/gif"
Add a magic byte to the file	Magic bytes function as signatures used by the web server to identify the type of file that is being uploaded. For example, when adding "GIF87a" to the beginning of the script, the server will think of it as a GIF file.
Try reducing the file size	If a file size restriction is being used, a smaller script can be uploaded to gain remote code execution, such as <code>&lt;?php echo system(\$_REQUEST['cmd']); ?&gt;</code>
Try using executable extensions	Certain executable extensions may still be allowed, for example .shtml, .asa, .cer, ".asax", ".swf", or ".xap".
Add a null byte to the file name	If the site is using file extension whitelists, this can often be bypassed by adding %00 (HTML encoding) or \x00 (hex encoding) to the end of the file name. For example: php-reverse-shell.php%00.gif
Add special characters before file extension	In order webserver, adding special characters such as ;%\$& just after the file name, for example shell;.php can help bypass file extension whitelists
Insert EXIF data	An executable script can be inserted into an image in the form of a metadata comment, which will then be executed when the web server uses the image in a page
Try using Windows 8.3 notation for the file name	The Windows 8.3 short name version can be used in the file name. For example shell.aspx will become SHELL~1.ASP

Try finding characters that are converted to other useful characters during the file upload process.	For instance, when running PHP on IIS, the ">", "<", and double quote " characters respectively convert to "?", "*", and "." characters that can be used to replace existing files (e.g. "web<<" can replace the "web.config" file). In order to include the double quote character in the filename in a normal file upload request, the filename in the "Content-Disposition" header should use single quotes (e.g. filename='web"config' to replace the "web.config" file).
Try adding neutral characters after the filename	Special characters like spaces or dots in Windows or dots and slashes in a Linux at the end of a filename will be removed automatically (e.g. "shell.aspx ... ..", "script.asp ", or "file.asp."). Although slash or backslash characters are unlikely to succeed as they are normally used to separate directories, they are worth a try (e.g. "shell.php/" or "scri[t.php.\").

## Conclusion

File upload vulnerabilities are very common when conducting a penetration test against web applications, knowing how to bypass file restrictions is key as these will often result in a full system compromise.

## Sources & Additional Resources

[https://owasp.org/www-community/vulnerabilities/Unrestricted\\_File\\_Upload](https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload)

([https://owasp.org/www-community/vulnerabilities/Unrestricted\\_File\\_Upload](https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload))

[https://null-byte.wonderhowto.com/how-to/bypass-file-upload-restrictions-web-apps-get-shell-](https://null-byte.wonderhowto.com/how-to/bypass-file-upload-restrictions-web-apps-get-shell-0323454/)

[0323454/](https://null-byte.wonderhowto.com/how-to/bypass-file-upload-restrictions-web-apps-get-shell-0323454/) (<https://null-byte.wonderhowto.com/how-to/bypass-file-upload-restrictions-web-apps-get-shell-0323454/>)

<https://www.exploit-db.com/docs/english/45074-file-upload-restrictions-bypass.pdf>

(<https://www.exploit-db.com/docs/english/45074-file-upload-restrictions-bypass.pdf>)

asp (<https://steflan-security.com/tag/asp/>)    aspx (<https://steflan-security.com/tag/aspx/>)

bypass (<https://steflan-security.com/tag/bypass/>)

Cheat Sheet (<https://steflan-security.com/tag/cheat-sheet/>)

cheatsheet (<https://steflan-security.com/tag/cheatsheet/>)

file upload (<https://steflan-security.com/tag/file-upload/>)

Hacking (<https://steflan-security.com/tag/hacking/>)

Penetration Testing (<https://steflan-security.com/tag/penetration-testing/>)

Pentesting (<https://steflan-security.com/tag/pentesting/>)

php (<https://steflan-security.com/tag/php/>)

restriction (<https://steflan-security.com/tag/restriction/>)