# Cross Site Scripting

## XSS

XSS means an attacker is able to get a victim to execute attacker-controlled Javascript in the origin of a thirdparty web page

## In the web Javascript always comes via HTML

```
<script src="//example.org/script.js"></script>
<img src=x onerror=alert(1)>
```

## we use that to exploit things

## Common XSS

```
<h1>Hello World</h1>
<p><?php echo $_GET['var1']; ?></p>
```

Usually we have an existing HTML document that dynamically includes data that we can control

But there's another way:

If we can convince the server to deliver a whole HTML document we also have XSS

# File Upload XSS

## How does a browser decide what's an HTML document?

## The Content-Type header

```
Content-Type: text/html; charset=UTF-8
```

| Dangerous | Safe (usually) |
|---|---|
| text/html | text/plain |
| text/xml | image/jpg |
| application/xml | application/octet-stream |
|  | bogus/contenttype |

## Let's look at an attack

# XSS using quirky implementations of ACME http-01

September 4, 2018

# ACME http-01 validation

*http://example.org/.well-known/acme-challenge/TOKEN1*

**Response:** *TOKEN1.TOKEN2*

Some implementations reflect TOKEN1:

*http://example.org/.well-known/acme-challenge/[anything]*

**Response:** *[anything].TOKEN2*

This looks like a classic reflected XSS, except...

this is not an HTML document

*MIME that tries to figure out the content-type depending on the first bytes of the response. [...] For example <b> would lead to content type text/html [...]*

## This does not sound good

## There are probably more problems with that

## Apache mod_mime_magic

It's a module that enables XSS attacks by introducing MIME Sniffing on the server

# Apache mod_mime_magic

*This module determines the MIME type of files in the same way the Unix file(1) command works: it looks at the first few bytes of the file.*

*Apache documentation*

## mod_mime_magic parser

Parser code is based on an old fork of the "file" utility

## Some 90s style C code is parsing files

## Memory corruption?

Probably...

(the code is not testing friendly)

## Apache with mod_mime_magic

- If file extension is in */etc/mime.types* use that for the *Content-Type* header
- Else try to guess it based on the file content

## If we have...

- a web application that allows file uploads
- with an unusual file type not in /etc/mime.types

we have XSS, because we can upload HTML and mod_mime_magic will autodetect it and set the *Content-Type* accordingly

# /etc/mime.types

```
text/html                       htm html shtml
image/jpeg                      jpe jpeg jpg
model/iges                      iges igs
```

## */etc/mime.types*

Not standardized at all, every operating system and every Linux distribution maintains its own variation

## This *mod_mime_magic* is dangerous, probably we should just disable it

## Good idea, as long as you're not using shared hosting

*mod_mime_magic* can't be disabled via *.htaccess* or per virtual host, only for the whole Apache httpd server

## But if we disable it server-wide we're good, right?

## Not so fast…

## What happens if the web server can't identify the MIME type?

It just won't send a *Content-Type* header

## Browsers also do MIME Sniffing

If you send a browser an HTML document without a *Content-Type* header it will render it

# Same attack works just as well without mod_mime_magic

But there's a security header to save us:

***X-Content-Type-Options: nosniff***

*The X-Content-Type-Options response HTTP header is a marker used by the server to indicate that the MIME types advertised in the Content-Type headers should not be changed and be followed. This allows to opt-out of MIME type sniffing, or, in other words, it is a way to say that the webmasters knew what they were doing.*

*MDN / X-Content-Type-Options*

So we can set *X-Content-Type-Options: nosniff* and we're safe from these attacks

## Not so fast...

## You have to read the fine print

*Note: nosniff only applies to "script" and "style" types. Also applying nosniff to images turned out to be incompatible with existing web sites.*

*MDN / X-Content-Type-Options*

*X-Content-Type-Options: nosniff* applies to script and style files, it explicitly doesn't apply to images, but it says nothing about direct navigation

## *X-Content-Type-Options: nosniff* and direct navigation

Chrome: Will display text

Safari: Will download file

Firefox and Edge: Will sniff the MIME type and render HTML

## In both Firefox and Edge HTML will be sniffed despite *X-Content-Type-Options:nosniff*

Mozilla accepts that this is a bug and should be fixed in Firefox, but it hasn't happened yet

## Microsoft Edge

*We have completed our investigation, and the behavior that you reported has minimal impact and therefore does not meet our bar for servicing at this time.*

## If a web application

- allows uploads of unusual file types not in /etc/mime.types
- And the web server has mod_mime_magic
- Or the web application is not setting *X-Content-Type-Options: nosniff*
- Or the browser is Firefox or Edge

we have XSS

## Example: Wordpress

## A user with an "Author" role can upload media files

```
	$misc_exts          = array(
		// Images.
		'jpg', 'jpeg', 'png', 'gif',
		// Video.
		'mov', 'avi', 'mpg', '3gp', '3g2',
		// "audio".
		'midi', 'mid',
		// Miscellaneous.
		'pdf', 'doc', 'ppt', 'odt', 'pptx',
		'docx', 'pps', 'ppsx', 'xls',
		'xlsx', 'key',
	);
```

## *.3g2* Videos

Not in */etc/mime.types* on Debian/Ubuntu systems

Wordpress will try to detect the MIME type (with PHP's libmagic bindings) and will not accept the file if it doesn't match, but it will accept the file if the MIME type can't be detected

## We need a file that libmagic won't identify as any valid file type, but the browser will identify it as HTML

# Demo Wordpress

## Wordpress disclosure

- September 10th 2018: Reported to Wordpress via HackerOne
- Still not fixed

## Example Mailman/Pipermail

## Mailman 2.x comes with a tool called Pipermail to create mailing list archives

## Attachment handling

An attachment with an unknown extension or no extension will be renamed to attachment.obj

Mailman tries to detect HTML, but the same bypass works:

*<img src=x onerror=alert(1)><!---[+ binary garbage]*

## *.obj*

Not in mime.types in Fedora, Redhat, Suse, Ubuntu, Debian, ...

Only Gentoo knows it (it's a tgif file)

## XSS for almost all Mailman mailing lists with a public archive

Mailman changed the default extension from .obj to .bin, which is served as application/octet-stream by all major Linux distributions, but the fix hasn't been released yet

# Demo Mailman

## More vulnerable applications

- Joomla / .xcf on Fedora (mitigated in 3.9.3, CVE-2019-7742)
- Vanilla forum / .fla files (still unfixed)

## What can we do?

## Can we prevent these attacks by always sending a default MIME type?

## Let's set a safe MIME type (e.g. text/plain or application/octet-stream) for every unknown file extension

# Apache "DefaultType" Directive

Has been removed in Apache 2.4

## WHY???

## W3C Standard Authoritative Metadata

A standard to enable Cross Site Scripting.

Setting a default MIME type would prevent these attacks, but people writing standards really don't like it nginx sends application/octet-stream by default

## File extension quirks

Web servers usually decide based on the file extension which MIME type to use

```
if ((pathinfo($_FILES['up']['name'], PATHINFO_EXTENSION) == 'jpg') ||
   (pathinfo($_FILES['up']['name'], PATHINFO_EXTENSION) == 'png') ||
   (pathinfo($_FILES['up']['name'], PATHINFO_EXTENSION) == 'gif')) {
    die("Error: Only images allowed!)";
} else {
   // copy uploaded file to upload dir
   [...]
}
```

## Is this code safe from XSS?

## What happens if we upload a file named ".jpg"?

## .jpg

What's the file extension?

## PHP and Apache disagree

```
/* [...]
 * Leading dots are considered to be part of the base name (a file named
 * ".png" is likely not a png file but just a hidden file called png).
 */
```

*Apache mod_mime.c*

## .jpg

PHP's pathinfo() function returns "jpg", but Apache will serve it without a mime type

A file named ".jpg" (or .[extension]) will pass a check on the file extension in PHP, but Apache will serve it without a MIME type, leading to XSS

# What's the extension of ".jpg"?

| Languages | empty | .jpg |
|---|---|---|
| PHP | - | ✓ |
| Ruby | ✓ | - |
| Python | ✓ | - |
| Web servers | empty | .jpg |
| Apache | ✓ | - |
| Nginx | - | ✓ |
| Caddy | ✓ | - |

# Internet Explorer

## Can we get MIME Sniffing if the server sends a file with a non-HTML MIME type like text/plain?

# [0day] Text/Plain Considered Harmful

admin

April 18, 2017

0day, content-type, exploit, ie, vulnerability

Hello reader!

It is time for another blogpost! This time it is about a bug I found and I believe it could be quite useful for you someday. It is worth mentioning it affects all versions of IE (tested on win 7, win 8.1 and win 10). It does not affect Edge.

## Internet Explorer supports displaying HTML emails in .eml format (RFC 822)

# .eml

```
TESTEML
Content-Type: text/html

<iframe src="https://example.org/test.txt"></iframe>
```

Iframes within .eml files ignore the Content-Type and will be MIME-sniffed

This blogpost is from 2017 and it works in the latest version of Internet Explorer 11

*Internet Explorer 11 is the last version of Internet Explorer, and will continue to receive security updates, compatibility fixes, and technical support on Windows 7, Windows 8.1, and Windows 10.*

*Microsoft*

I don't think that's true

# File upload MIME type

## Browsers will send a MIME type with file uploads

Obviously this is user-supplied data and can't be trusted

It's possible that this allows attacks if a web application trusts the supplied MIME type

## Caddy Web Server

*its security defaults and unparalleled usability*

## Caddy does MIME Sniffing by default, it can't be switched off

Caddy developer told me they just use Go's functionality *go http: ServeContent*

# Defense

## Browsers

Please fix **X-Content-Type-Options:nosniff**

## Make a long term plan plan to deprecate MIME sniffing

# Web servers

Don't do server side MIME sniffing

# Apache HTTPD administrators

Disable mod_mime_magic

## Operating system vendors and Linux distributions

Consider adding additional file types (.3g2, .obj, .fla, .xcf) to */etc/mime.types*

## Please standardize */etc/mime.types* across distributions and operating systems

## Web applications / webmasters

## The safest option is a sandbox domain for file uploads, but it's often impractical

## Always set **X-Content-Type-Options: nosniff**

(even though it's imperfect)

## If you support any unusual file types consider forcing the MIME type

## Reject file uploads starting with a dot

## Summary

MIME Sniffing is dangerous and needs to go away

# Don't sniff the mime

Any questions?

hboeck.de @hanno