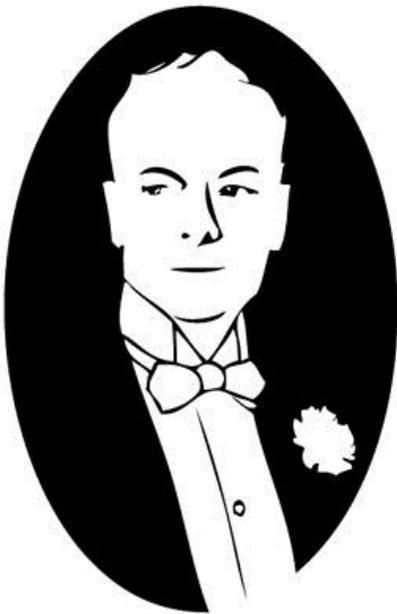


Advanced Information Security

Corporation



27/02/2014

Advanced Information Security Corporation Security Advisory Report

Information Security Expert, Nicholas Lemonias

Google's Unrestricted File Upload Vulnerability

Web Application Security

Domain: [YouTube.com](https://www.youtube.com)

Type: Web Application Vulnerability

Affected Service: YouTube/Upload

Vulnerability: Unrestricted File Upload

Date: 27/02/2014

Vendor Overview

Google is an American multinational corporation specializing in Internet-related services and products. These include search, cloud computing, software, and online advertising technologies. Google was founded by Larry Page and Sergey Brin while they were Ph.D. students at Stanford University. They incorporated Google as a privately held company on September 4, 1998. An initial public offering followed on August 19, 2004. Its mission statement from the outset was "to organize the world's information and make it universally accessible and useful", and its unofficial slogan was "Don't be evil".

Service Overview

YouTube is a video-sharing website, created by three former PayPal employees in February 2005 and owned by Google since late 2006, on which users can upload, view and share videos. The company is based in San Bruno, California, and uses Adobe Flash Video and HTML5 technology to display a wide variety of user-generated video content, including video clips, TV clips, and music videos, and amateur content such as video blogging, short original videos, and educational videos. Most of the content on YouTube has been uploaded by individuals, but media corporations including CBS, the BBC, Vevo, Hulu, and other organizations offer some of their material via YouTube, as part of the YouTube partnership program. Unregistered users can watch videos, and registered users can upload an unlimited number of videos.. YouTube, LLC was acquired by Google for US\$1.65 billion in November 2006 and now operates as a Google subsidiary.

Description

A security report was made to Google Inc. on the 26th of February, in reference to Google's coordinated security reward program. The security issue presented, allowed circumvention of web-based control handlers used by the YouTube API, which determined the file-types permitted to be written on YouTube's Storage network. The validation occurred at the application-layer, through a web-based form; Therefore a user could tamper with the Http data, in order to bypass any web-based file-type validation checks, and consequently to upload, any file of choice to the remote network. According to subject matter literature once the information security flow or scope, and or function of a design is circumvented, that constitutes to a security violation [5].

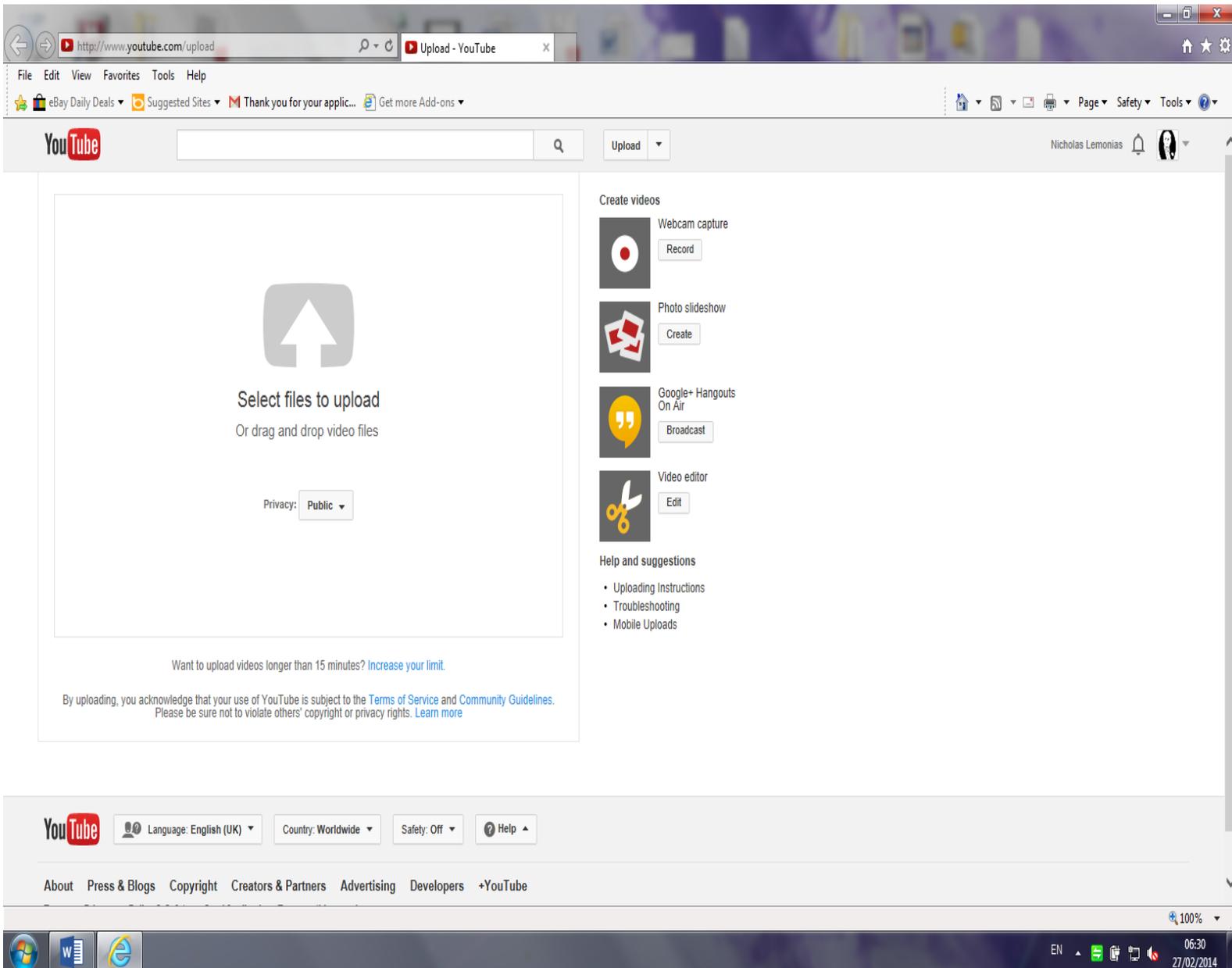
Responsible Disclosure Timeline

26th of February, 2014– Vendor Notification.

27th of February, 2014 – Problem Mitigation.

Appendices

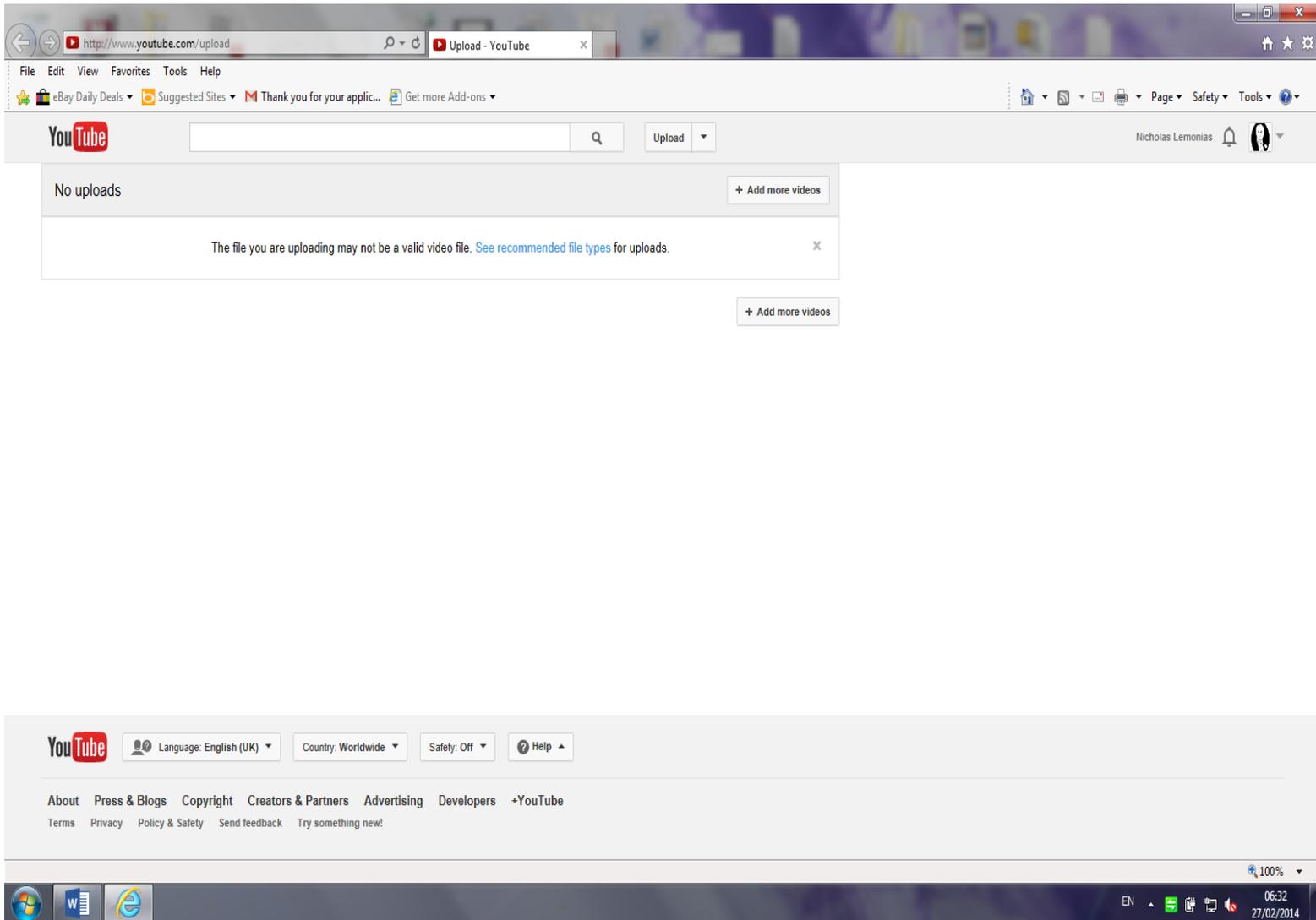
Proof of Concept Image 1



Description

Google's YouTube Upload service suffered from an unrestricted file upload vulnerability.

Proof of Concept Image 2



Description

Any attempt to upload an unsupported file-type, the application normally returns the above error message. Namely, that only media files could be uploaded to the remote server and prohibits any write requests to the server.

Proof of Concept Image 3

The screenshot displays the Tamper Data tool interface. A modal window titled "Tamper Popup" shows the details of an ongoing request to `http://www.youtube.com/yummy_ajax`. The request is a GET method with a status of "pending".

Request Header Name	Request Header Value	Post Parameter Name	Post Parameter Value
Host	www.youtube.com	session_token	uJlAvgTH7Vv/x4LmpEKsIM8X4FV8M
User-Agent	Mozilla/5.0 (Windows; U; Windows	frontend_id	web_upload%3A6e17862b-d475-4df7
Accept	text/xml,application/xml,application/	version	1
Accept-Language	en-us,en;q=0.5	filename	unsupportedfile.zip
Accept-Encoding	gzip,deflate		
Accept-Charset	ISO-8859-1,utf-8;q=0.7,*;q=0.7		
Keep-Alive	300		
Connection	keep-alive		
X-YouTube-Page-CL	61962321		
X-YouTube-Page-Timestamp	Fri Feb 21 12:32:29 2014 (139301474)		
Content-Type	application/x-www-form-urlencoded		
Referer	http://www.youtube.com/upload		
Content-Length	231		
Cookie	YSC=gMrMam1kyQQ; VISITOR_INFO_LIVE		

The background shows a list of ongoing requests with columns for Time, Duration, Total Duration, Size, Method, Status, Content Type, URL, and Load Flags. The URL column shows various requests to `http://www.youtube.com/` and `https://apis.google.com/`.

Description

The above example demonstrates tampering of the application's security controls through an HTTP parser. Successful exploitation of http headers, results to arbitrary file submission. In our test we could submit any file of our choice. In the above example, `unsupportedfile.zip` was successfully accepted by the application, therefore surpassing all web-based file-type validation checks [1], [2], [5].

Proof of Concept Image 4

The screenshot shows the Tamper Data - Ongoing requests window. The main window displays a list of requests with columns for Time, Duration, Total Duration, Size, Method, Status, Content Type, URL, and Load Flags. A 'Tamper Popup' dialog box is open, showing the details of a request to `http://www.youtube.com/upload/rupio?authuser=0`.

Tamper Popup Request Details:

Request Header Name	Request Header Value	Post Parameter Name
Host	www.youtube.com	
User-Agent	Mozilla/5.0 (Windows; U; Windows N	
Accept	text/xml,application/xml,application	
Accept-Language	en-us,en;q=0.5	
Accept-Encoding	gzip,deflate	
Accept-Charset	ISO-8859-1,utf-8;q=0.7,*;q=0.7	
Keep-Alive	300	
Connection	keep-alive	
X-GUploader-Client-Info	ty silverlight; clientVersion=61610330	
Content-Type	application/x-www-form-urlencoded	
Referer	http://www.youtube.com/upload	
Content-Length	1615	
Cookie	YSC=gMrMam1kyQQ; VISITOR_INFO_LIVE	

The background window shows a list of requests with various content types and load flags, including `application/x-unknown-content-type`, `application/javascript`, and `application/x-unknown-content-type`.

Description

The above example demonstrates the response headers and the mixed content-types allowed by the application [3], [5].

Proof of Concept Image 5

Tamper Data - Ongoing requests

Start Tamper Stop Tamper Clear Options Help

Filter Show All

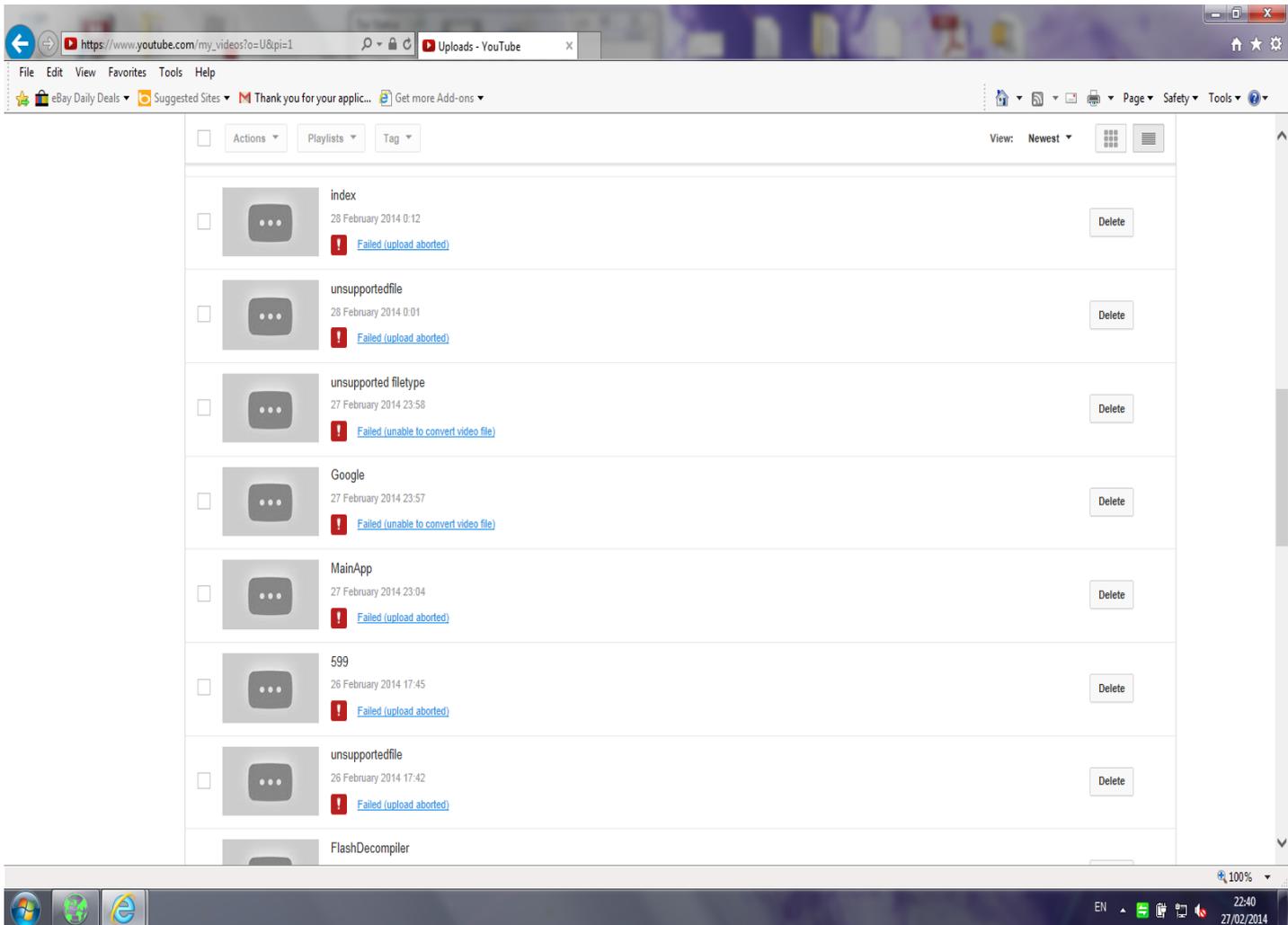
Time	Duration	Total Duration	Size	Method	Status	Content Type	URL	Load Flags
15:58:33.007	172 ms	172 ms	0	GET	204	text/html	http://www.youtube.com...	LOAD_NORMAL
15:58:33.101	0 ms	0 ms	unknown	GET	pending	unknown	http://www.onepick-ope...	LOAD_NORMAL
15:58:36.080	220 ms	220 ms	0	GET	204	text/html	http://www.youtube.com...	LOAD_NORMAL
15:58:38.363	125 ms	125 ms	-1	GET	304	application/x-unknown-content-type	http://www.youtube.com...	LOAD_NORMAL
15:58:40.235	125 ms	125 ms	0	GET	204	text/html	http://www.youtube.com...	LOAD_NORMAL
15:59:50.872	1248 ms	1248 ms	0	GET	204	text/html	http://www.youtube.com...	LOAD_NORMAL
16:01:24.466	312 ms	312 ms	0	POST	405	text/html	http://www.youtube.com...	LOAD_BYPASS_CACHE LOAD_B...
16:01:38.724	780 ms	780 ms	985	POST	405	text/html	http://www.youtube.com...	LOAD_BYPASS_CACHE LOAD_B...
16:01:51.844	390 ms	390 ms	119	GET	200	application/json	http://www.youtube.com...	LOAD_BACKGROUND
16:02:39.954	281 ms	281 ms	0	POST	405	text/html	http://www.youtube.com...	LOAD_BYPASS_CACHE LOAD_B...
16:03:09.017	47 ms	47 ms	1461	GET	405	text/html	http://www.youtube.com...	LOAD_NORMAL
16:03:14.571	124 ms	124 ms	0	GET	204	text/html	http://www.youtube.com...	LOAD_NORMAL
16:03:43.212	219 ms	219 ms	748	POST	200	text/html	http://www.youtube.com...	LOAD_BYPASS_CACHE LOAD_B...
16:03:51.933	140 ms	140 ms	0	GET	204	text/html	http://www.youtube.com...	LOAD_NORMAL
16:03:57.533	125 ms	125 ms	0	GET	204	text/html	http://www.youtube.com...	LOAD_NORMAL
16:04:05.458	234 ms	234 ms	304	GET	302	text/html	http://fls.doubleclick.net/...	LOAD_DOCUMENT_URI
16:04:10.013	218 ms	218 ms	177	GET	200	text/html	http://2542116.flis.doublec...	LOAD_DOCUMENT_URI LOAD_...
16:04:15.598	140 ms	140 ms	0	GET	204	text/html	http://www.youtube.com...	LOAD_NORMAL

Request Header Name	Request Header Value	Response Header Name	Response Header Value
Host	www.youtube.com	Status	OK - 200
User-Agent	Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.8.1.20) Gecko/20081217 Firefox/2.0.0.20	Set-Cookie	SID=DQAAANYAAADyhlnyfm97RAYyRq1ZJ59FkrdC9ptoSib3Sbj-XOSpaSFlwk7IMZ6F-814nEjmHH3OVFTMzkFnEX-yX...
Accept	text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5	Content-Type	text/html; charset=utf-8
Accept-Language	en-us,en;q=0.5	X-GUploader-Stats-URL	https://clients2.google.com/uploadstats
Accept-Encoding	gzip,deflate	Location	http://upload.youtube.com/?authuser=0&upload_id=AEnB2Uo9k6NWfs_CQlpw94yxG3VojG8P3lc3A4IPj9hTqj0XkFk...
Accept-Charset	ISO-8859-1,utf-8;q=0.7,*;q=0.7	Content-Length	748
Keep-Alive	300	Date	Wed, 26 Feb 2014 16:03:42 GMT
Connection	keep-alive	Server	HTTP Upload Server Built on Feb 24 2014 14:50:26 (1393282226)
X-GUploader-Client-Info	mechanism=scotty silverlight; clientVersion=61610330	Expires	Wed, 26 Feb 2014 16:03:42 GMT
Referer	http://www.youtube.com/upload	Cache-Control	private
Content-Length	1615	Alternate-Protocol	80:quic
Cookie	YSC=gMrMamLkyQQ; VISITOR_INFO_LIVE=nx5jIMOeZY; SID=DQAAANYAAADyhlnyfm97RAYyRq1ZJ59FkrdC9ptoS...		
Pragma	no-cache		
Cache-Control	no-cache		
POSTDATA	{ "protocolVersion": "0.8", "createSessionRequest": { "fields": { "external": { "name": "file", "filename": "unsupportedfile.zip", "f...		

Description

The executable file has successfully been uploaded to YouTube's storage network and our request was successful. Any firewall protections would also be invaluable in such cases.

Proof of Concept Image 6



Description

The YouTube Video Manager, confirms our uploads. It is pertinent to note that this report has not confirmed remote file execution.

References

- [1] Dalton, Michael, Christos Kozyrakis, and Nikolai Zeldovich. "Nemesis: Preventing Authentication & Access Control Vulnerabilities in Web Applications." *USENIX Security Symposium*. 2009.
- [2] Microsoft Corporation. (2014). *Improving Web Application Security: Threats and Countermeasures*.

Available: <http://msdn.microsoft.com/en-us/library/ms994921.aspx>. Last accessed 2014.

[3] Microsoft Corporation. (2014). *Securing Sites with Web Site Permissions*. Available: [http://technet.microsoft.com/en-us/library/cc756133\(WS.10\).aspx](http://technet.microsoft.com/en-us/library/cc756133(WS.10).aspx) . Last accessed 2014.

[4]OWASPFoundation.(2014). *Open Web Application Security Project - Unrestricted File Upload*. Available: https://www.owasp.org/index.php/Unrestricted_File_Upload. Last accessed 2014.

[5] Saltzer, Jerome H., and Michael D. Schroeder. "The protection of information in computer systems." *Proceedings of the IEEE* 63.9 (1975): 1278-1308.