

Are HTTPS URLs encrypted?

Asked 15 years, 10 months ago Modified 6 months ago

Viewed 420k times



1333

Are all URLs encrypted when using TLS/SSL (HTTPS) encryption? I would like to know because I want all URL data to be hidden when using TLS/SSL (HTTPS).



If TLS/SSL gives you total URL encryption then I don't have to worry about hiding confidential information from URLs.



ssl

https

httprequest

Share

Improve this question

Follow

edited Apr 29, 2019 at 18:00



Miyagi

151 ● 2 ● 17

asked Jan 31, 2009 at 21:15




Daniel Kivatinos

25k ● 23 ● 63 ● 81

92 It's probably a bad idea to put confidential data in the URL anyway. It will be displayed in the browser's address bar too, remember? People don't like it if their password is visible to anyone who happens to glance at the screen. Why do you think you need to put confidential data in the URL?
– [Stack Overflow is garbage](#) Jan 31, 2009 at 22:03

57 URLs are also stored in browser history and server logs - if I wanted to have my name and password stored somewhere, it would not be in these two places. – [Piskvor left the building](#) Jun 30, 2010 at 15:33

59 For example, suppose I visit `https://somewhere_i_trust/ways_to_protest_against_the_government/`. Then the URL contains confidential data, namely the suggestion that I am considering protesting against my government. – [Steve Jessop](#) Sep 26, 2011 at 8:42 

51 I was asking myself this question when making an HTTP request from a native (not browser based) App. I'm guessing this may interest mobile App developers. In this case, the comments above (while true) are irrelevant (no url visible, no browsing history), making the answer, to my understanding a simple: "Yes, it's encrypted". – [DannyA](#) Jun 18, 2012 at 18:11

35 For those who think once you are HTTPS no one knows where you're going, **read this first:** The hostname of the server (e.g. example.com) **will still be leaked due to [SNI](#)**. This has absolutely nothing to do with DNS and the leak will occur even if you don't use DNS or use encrypted DNS. – [Pacerier](#) Nov 9, 2015 at 21:38

14 Answers

Sorted by:

Highest score (default)



1169



Yes, the SSL connection is between the TCP layer and the HTTP layer. The client and server first establish a secure encrypted TCP connection (via the SSL/TLS protocol) and then the client will send the HTTP request (GET, POST, DELETE...) over that encrypted TCP connection.



[Note however](#) (as also noted in the comments) that the **domain name** part of the URL is sent in clear text during the first part of the TLS negotiation. So, the domain name of the server can be sniffed. But not the rest of the URL.

Share Improve this answer

edited Jul 10, 2022 at 19:43

Follow



matteo

3,068 ● 7 ● 46 ● 65

answered Jan 31, 2009 at 21:17



Marc Novakowski

45.4k ● 11 ● 61 ● 63

127 It is still worth noting the thing mentioned by @Jalf in the comment on the question itself. URL data will also be saved in the browser's history, which may be insecure long-term.

– [Michael Ekstrand](#) Jul 12, 2009 at 1:37

24 Not just GET or POST. Can also be DELETE, PUT, HEAD, or TRACE. – user142019 Mar 2, 2011 at 22:13

9 Yes it could be a security issue for a browser's history. But in my case I'm not using browser (also the original post did not mention a browser). Using a custom https call behind the scenes in a native app. It's a simple solution to making sure your app's sever connection is secure. – [zingle-dingle](#) May 28, 2013 at 23:12

44 Note however that the DNS resolve of the URL is probably not encrypted. So someone sniffing your traffic could still probably see the domain you're trying to access.

– [ChewToy](#) Jun 19, 2013 at 7:35

26 SNI breaks the 'host' part of SSL encryption of URLs. You can test this yourself with wireshark. There is a selector for



1036

Since nobody provided a wire capture, here's one.

Server Name (the domain part of the URL) is presented in the `ClientHello` packet, in **plain text**.



The following shows a browser request to:

`https://i.stack.imgur.com/path/?`



`some=parameters&go=here`



141	6.25039500	188.25.154.112	104.16.111.18	TLSv1.2	579 Client Hello
142	6.26535000	151.101.193.69	188.25.154.112	TLSv1.2	398 Application Data
143	6.26580700	151.101.193.69	188.25.154.112	TLSv1.2	412 Application Data
144	6.28061300	104.16.111.18	188.25.154.112	TCP	62 443->10970 [ACK] Seq=1 Ack=518 win=30720 Len=0

Compression Methods (1 method)
Extensions Length: 401
Extension: renegotiation_info
Type: renegotiation_info (0xff01)
Length: 1
Renegotiation Info extension
Extension: server_name
Type: server_name (0x0000)
Length: 22
Server Name Indication extension
Server Name list length: 20
Server Name Type: host_name (0)
Server Name length: 17
Server Name: i.stack.imgur.com

0000 fc e3 3c 9c eb 8f 00 22 4d 7c 7a 52 88 64 11 00 .<... M|zR.d..
0010 1d 59 02 2f 00 21 45 00 02 2d 0e 81 40 00 80 06 .Y./!E. -.@..
0020 bc 9d bc 19 9a 70 68 10 6f 12 2a d9 01 bb ad d6 .?f(CP.
0030 bf a4 3f 66 28 63 50 18 01 02 93 ee 00 00 16 033.op..
0040 01 02 00 01 00 01 fc 03 03 b1 33 c3 6f 70 fb fb-h...V=
0050 60 17 1e f7 a7 3d 86 2d 01 68 f2 fd d8 19 56 3d .V.J.: .Ft.]..
0060 18 27 16 56 8d 4a fd 3a e6 20 46 74 1d 5d b9 8a\$.i. .@.-:
0070 ed a7 9d 13 d9 24 69 01 d2 c9 c7 40 a1 93 2d 3a ...E.... [&." +/..
0080 16 a1 04 45 0e fb c8 ff 5b 26 00 22 c0 2b c0 2f ...0.... /.5..
0090 c0 2c c0 30 cc a9 cc a8 cc 14 cc 13 c0 09 c0 13
00a0 c0 0a c0 14 00 9c 00 9d 00 2f 00 35 00 0a 01 00
00b0 01 91 ff 01 00 01 00 00 00 00 16 00 14 00 00 11
00c0 69 2e 73 74 61 63 6b 2e 69 6d 67 75 72 2e 63 6f i.stack.imgur.co
00d0 6d 00 17 00 00 00 23 00 c0 27 0c 0d e3 03 eb c5#.....
00e0 04 81 6a c8 8d b8 fc b0 3c f0 c9 b5 7a 4a fb dcS...23...

TLS RECORD LAYER

[See this answer](#) for more on TLS version fields (there are 3 of them - not versions, fields that each contain a version number!)

From <https://www.ietf.org/rfc/rfc3546.txt>:

3.1. Server Name Indication

[TLS] does not provide a mechanism for a client to tell a server the name of the server it is contacting. It may be desirable for clients to

provide this information to facilitate secure connections to servers that host multiple 'virtual' servers at a single underlying network address.

In order to provide the server name, clients MAY include an extension of type "server_name" in the (extended) client hello.

In short:

- FQDN (the domain part of the URL) **MAY** be transmitted **in clear** inside the `clientHello` packet if SNI extension is used
- The rest of the URL (`/path/?some=parameters&go=here`) has no business being inside `clientHello` since the request URL is a HTTP thing (OSI Layer 7), therefore it will never show up in a TLS handshake (Layer 4 or 5). That will come later on in a `GET /path/?some=parameters&go=here HTTP/1.1` HTTP request, **AFTER** the **secure** TLS channel is established.

EXECUTIVE SUMMARY

Domain name MAY be transmitted in clear (if SNI extension is used in the TLS handshake) but URL (path and parameters) is always encrypted.

MARCH 2019 UPDATE

Thank you [carlin.scott](#) for bringing this one up.

The payload in the SNI extension can now be encrypted via [this draft RFC proposal](#). This capability only exists in TLS 1.3 (as an option and it's up to both ends to implement it) and there is no backwards compatibility with TLS 1.2 and below.

CloudFlare is doing it and you can read more about the internals here — [If the chicken must come before the egg, where do you put the chicken?](#)

In practice this means that instead of transmitting the FQDN in plain text (like the Wireshark capture shows), it is now encrypted.

NOTE: This addresses the privacy aspect more than the security one since a reverse DNS lookup MAY reveal the intended destination host anyway.

SEPTEMBER 2020 UPDATE

There's now a draft RFC for encrypting the entire Client Hello message, not just the SNI part:

https://datatracker.ietf.org/doc/draft-ietf-tls-esni/?include_text=1

At the time of writing this browser support is VERY limited.

Share Improve this answer

edited Jun 4 at 18:17

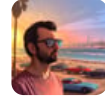
Follow



corn on the cob

2,284 ● 3 ● 22 ● 34

answered Aug 2, 2016 at 18:26



evilSnobu

26.2k ● 8 ● 45 ● 73

81 Perfect answer, with complete explanation from A to Z. I love the Executive summary. Made my day @evilSnobu
– [oscaroscar](#) Apr 18, 2018 at 14:04

4 Perfect answer, upvote! Still consider the client part since the browser history may be a leak. However, regarding the transport layer, URL-parameters are encrypted.
– [Jens Kreidler](#) May 7, 2018 at 6:51

2 You may want to update this answer with the fact that TLS 1.3 encrypts the SNI extension, and the biggest CDN is doing just that: blog.cloudflare.com/encrypted-sni Of course a packet sniffer could just do a reverse-dns lookup for the IP addresses you're connecting to. – [carlin.scott](#) Mar 21, 2019 at 18:00

1 @evilSnobu, but username:password part of [username:password@domain.com](#) is encrypted, right? So it's secure to pass sensitive data in url using https.
– [Maksim Shamihulau](#) Nov 11, 2019 at 14:47

1 They are encrypted on the wire (in transport) but if either end (user or server) logs the URL to a plain text file and does not sanitize credentials... now that's a different conversation.
– [evilSnobu](#) Nov 11, 2019 at 16:32 ✎



As the [other answers](#) have already pointed out, https "URLs" are indeed encrypted. However, your DNS

178



request/response when resolving the domain name is probably not, and of course, if you were using a browser, your URLs might be recorded too.



Share Improve this answer



Follow

edited May 23, 2017 at 12:18



Community Bot

1 • 1

answered Jan 31, 2009 at 21:26



Zach Scrivena

29.5k • 12 • 65 • 73

23 And URL recording is important since there are Javascript hacks that allow a completely unrelated site to test whether a given URL is in your history or not. You can make a URL unguessable by including a longish random string in it, but if it's a public URL then the attacker can tell that it has been visited, and if it has a short secret in it, then an attacker could brute-force that at reasonable speed. – [Steve Jessop](#) Sep 26, 2011 at 8:38

8 @SteveJessop, please provide a link to "Javascript hacks that allow a completely unrelated site to test whether a given URL is in your history or not" – [Pacerier](#) Mar 15, 2014 at 10:26

8 @Pacerier: hacks date of course, but what I was talking about at the time was things like stackoverflow.com/questions/2394890/.... It was a big deal back in 2010 that these issues were being investigated and the attacks refined, but I'm not really following it at the moment. – [Steve Jessop](#) Mar 15, 2014 at 10:49

3 @Pacerier: more examples: webdevwonders.com/..., webdevwonders.com/... – [Steve Jessop](#) Mar 15, 2014 at 11:19

- 1 You can use OpenDNS with it's encrypted DNS service. I use it on my Mac, but I found the Windows version not working properly. That was a while ago though, so it might work OK now. For Linux nothing yet.
- opendns.com/about/innovations/dnscrypt – SPRBRN Apr 22, 2014 at 15:02 ✎
-



I agree with the previous answers:

114

To be explicit:



With TLS, the first part of the URL (<https://www.example.com/>) is still visible as it builds the connection. The second part (/herearemygetparameters/1/2/3/4) is protected by TLS.



However there are a number of reasons why you should not put parameters in the GET request.

First, as already mentioned by others: - leakage through browser address bar - leakage through history

In addition to that you have leakage of URL through the http referer: user sees site A on TLS, then clicks a link to site B. If both sites are on TLS, the request to site B will contain the full URL from site A in the referer parameter of the request. And admin from site B can retrieve it from the log files of server B.)

Share Improve this answer

edited Jul 28, 2013 at 7:09

Follow



Mihai Maruseac

21.4k ● 7 ● 59 ● 110


answered Jul 28, 2013 at 6:49



Tobias

1,165 ● 1 ● 7 ● 2

-
- 3 @EJP You didn't understand what Tobias is saying. He's saying that if you click a link on site A that will take you to site B, then site B will get the referrer URL. For example, if you are on siteA.com?u=username&pw=123123, then siteB.com (which is linked to on the page of siteA.com) will receive "siteA.com?u=username&pw=123123" as the referring URL, sent to siteB.com inside the HTTPS by the browser. If this is true, that's very bad. Is this true Tobias? – [trusktr](#) Jun 26, 2014 at 18:37
-
- 9 @EJP, the domain is visible because of [SNI](#) which all modern web browsers use. Also see [this diagram](#) from the EFF showing that anyone can see the domain of the site you are visiting. This isn't about browser visibility. It's about what is visible to eavesdroppers. – [Buge](#) Dec 13, 2014 at 16:10
-
- 11 @trusktr: Browsers should not send a Referer header from HTTPS pages. This is [part of the HTTP specification](#). – [Martin Geisler](#) Aug 6, 2015 at 14:38
-

- 9 @MartinGeisler, Keyword is "should". Browsers don't much care about "should" (as opposed to "must"). From your own link: *"strongly recommended that the user be able to select whether or not the Referer field is sent. For example, a browser client could have a toggle switch for browsing openly/anonymously, which would respectively **enable** /disable the sending of Referer and From information"*. Ops, which is exactly what Chrome did. Except Chrome leaks the Referrer **even if you are in incognito mode**. – [Pacerier](#) Nov 9, 2015 at 21:34 
-



110



Entire request and response is encrypted, including URL.

Note that when you use a HTTP Proxy, it knows the address (domain) of the target server, but doesn't know the requested path on this server (i.e. request and response are always encrypted).



Share Improve this answer

Follow

edited Apr 23, 2014 at 9:59



[DanMan](#)

11.5k ● 4 ● 42 ● 61

answered Jan 31, 2009 at 21:17



[Peter Štibrný](#)

32.9k ● 16 ● 92 ● 116



Yes and no.

56





The server address portion is NOT encrypted since it is used to set up the connection.

This may change in future with encrypted SNI and DNS but as of 2018 both technologies are not commonly in use.

The path, query string etc. are encrypted.

Note for GET requests the user will still be able to cut and paste the URL out of the location bar, and you will probably not want to put confidential information in there that can be seen by anyone looking at the screen.

Share Improve this answer

Follow

edited Jul 18, 2018 at 10:01



[mikemaccana](#)

122k ● 110 ● 424 ● 528

answered Jan 31, 2009 at 21:20



[SoapBox](#)

20.6k ● 3 ● 54 ● 87

-
- 8 Would like to +1 this, but I find the "yes and no" misleading - you should change that to just point out that the server name

will be resolved using DNS without encryption.

– [L. Cornelius Dol](#) Jan 31, 2009 at 22:11

8 In my understanding, the OP uses the word URL in the right sense. I think this answer is more misleading, as it doesn't clearly make the difference between the hostname in the URL and the hostname in the DNS resolution. – [Guillaume](#) Nov 2, 2010 at 14:17

4 The URL is encrypted. Every aspect of the HTTP transaction is encrypted. Not just 'everything else'. Period. -1.
– [user207421](#) Mar 26, 2013 at 9:37 ✎

4 @EJP but the DNS lookup **does** use what is at one point part of the URL, so to the non-technical person, the entire URL is not encrypted. The non-technical person who's merely using Google.com to look up non-technical things does not know where the data ultimately resides or how it is handled. The domain, which is part of the URL the user is visiting, is not 100% encrypted because I as the attacker can sniff which site he is visiting. Only the /path of a URL is inherently encrypted to the layman (it doesn't matter how). – [trusktr](#) Jun 26, 2014 at 18:46 ✎

6 @EJP, @trusktr, @Lawrence, @Guillaume. All of you are mistaken. **This has nothing to do with DNS.** SNI "[send the name of the virtual domain as part of the TLS negotiation](#)", so even if you don't use DNS or if your DNS is encrypted, a sniffer can **still see the hostname** of your requests.
– [Pacerier](#) Nov 9, 2015 at 21:40 ✎



51



An addition to the helpful answer from Marc Novakowski - the URL is stored in the logs on the server (e.g., in /etc/httpd/logs/ssl_access_log), so if you don't want the server to maintain the information over the longer term, don't put it in the URL.



Share Improve this answer

answered Nov 2, 2010 at 14:03



Follow



Rhodri Cusack

619 ● 5 ● 3



16



It is now 2019 and the TLS v1.3 has been released. According to Cloudflare, the server name indication (SNI aka the hostname) can be encrypted thanks to TLS v1.3. So, I told myself great! Let's see how it looks within the TCP packets of cloudflare.com So, I caught a "client hello" handshake packet from a response of the cloudflare server using Google Chrome as browser & wireshark as packet sniffer. I still can read the hostname in plain text within the Client hello packet as you can see below. It is not encrypted.



*Ethernet						
Fichier Editer Vue Aller Capture Analyser Statistiques Telephonie Wireless Outils Aide						
ssl.handshake.extensions_server_name						
No.	Time	Source	Destination	Protocol	Length	Info
643	5.990151	2a01:cb18:188:7f00:...	2606:4700::c629:d6a2	TLSv1.3	591	Client Hello
646	5.991488	2a01:cb18:188:7f00:...	2606:4700::c629:d6a2	TLSv1.3	591	Client Hello
960	6.819933	2a01:cb18:188:7f00:...	2606:4700::6811:d109	TLSv1.3	591	Client Hello
1262	7.371428	192.168.1.203	93.184.220.178	TLSv1.3	571	Client Hello
1353	7.409341	192.168.1.203	93.184.220.178	TLSv1.3	577	Change Cipher Spec, Client Hello
2520	8.845429	2a01:cb18:188:7f00:...	2a00:1450:4007:80c:...	TLSv1.3	591	Client Hello
2694	9.120277	2a01:cb18:188:7f00:...	2606:4700::6811:7928	TLSv1.3	591	Client Hello
3915	11.340984	2a01:cb18:188:7f00:...	2606:4700::6811:164b	TLSv1.3	591	Client Hello

<div> <div>Server Name Indication extension</div> <div> <div>Server Name list length: 17</div> <div>Server Name Type: host_name (0)</div> <div>Server Name length: 14</div> <div>Server Name: cloudflare.com</div> </div> </div> <div> <div>Extension: extended_master_secret (len=0)</div> </div>	<pre> 00c0 00 00 00 00 00 13 00 11 00 00 63 6c 6f 75 64cloud 00d0 66 6c 61 72 65 2e 63 6f 6d 00 17 00 00 ff 01 00 flare.co m..... 00e0 01 00 00 0a 00 0a 00 08 9a 9a 00 1d 00 17 00 18 00f0 00 0b 00 02 01 00 00 23 00 00 00 10 00 0e 00 0c#..... 0100 02 68 32 08 68 74 74 70 2f 31 2e 31 00 05 00 05 ..h2http /1.1... 0110 01 00 00 00 00 00 0d 00 14 00 12 04 03 08 04 04 0120 01 05 03 08 05 05 01 08 06 06 01 02 01 00 12 00 0130 00 00 33 00 2b 00 29 9a 9a 00 01 00 00 1d 00 20 ...3+... 0140 2b 71 92 83 22 82 6a 06 6d 5a 96 ae 1f 94 7d f8 +q..."j mZ...} 0150 d8 ca e1 f8 b6 1c c7 a0 82 ae 8e 9d e0 3b c1 5b;-[0160 00 2d 00 02 01 01 00 2b 00 0b 0a 3a 3a 03 04 03+ 0170 03 03 02 03 01 00 1b 00 03 02 00 02 6a 6a 00 01jj.. 0180 00 00 15 00 ca 00 00 00 00 00 00 00 00 00 00 00 0190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 </pre>
--	--

So, beware of what you can read because this is still not an anonymous connection. A middleware application between the client and the server could log every domain that are requested by a client.

So, it looks like the encryption of the SNI requires additional implementations to work along with TLSv1.3

UPDATE June 2020: It looks like the Encrypted SNI is initiated by the browser. Cloudflare has a page for you to check if your browser supports Encrypted SNI:

<https://www.cloudflare.com/ssl/encrypted-sni/>

At this point, I think Google chrome does not support it. You can activate Encrypted SNI in Firefox manually. When I tried it for some reason, it didn't work instantly. I restarted Firefox twice before it worked:

Type: about:config in the URL field.

Check if network.security.esni.enabled is true. Clear your cache / restart

Go to the website, I mentioned before.

As you can see VPN services are still useful today for people who want to ensure that a coffee shop owner does not log the list of websites that people visit.

Share Improve this answer

edited Jun 21, 2020 at 7:47

Follow

answered Jan 24, 2019 at 22:47



Nicolas Guérinet

2,216 ● 1 ● 31 ● 41

1 "the SNI **can** be encrypted" - that's the key point. Checking cloudflare.com/ssl/encrypted-sni with current Google Chrome says "Your browser did not encrypt the SNI when visiting this page." It takes two to tango... – [Piskvor left the building](#) Sep 16, 2019 at 8:47 ✎

4 Apparently current Firefox *can* do ESNI, but it's disabled by default: you need to enable `network.security.esni.enabled`, set `network.trr.mode` to 2 (which currently sets your DoH resolver to CloudFlare), and restart the browser (sic!); then it will use ESNI - where supported by the domain's infrastructure. See blog.mozilla.org/security/2018/10/18/... for details. – [Piskvor left the building](#) Sep 16, 2019 at 9:10 ✎

Is this supposed to work with any modern DNS or is this a specific 3rd party (Cloudflare) solution? – [Louis Somers](#) May



11



A third-party that is monitoring traffic may also be able to determine the page visited by examining your traffic and comparing it with the traffic another user has when visiting the site. For example if there were 2 pages only on a site, one much larger than the other, then comparison of the size of the data transfer would tell which page you visited. There are ways this could be hidden from the third-party but they're not normal server or browser behaviour. See for example this paper from SciRate, <https://scirate.com/arxiv/1403.0297>.

In general other answers are correct, practically though this paper shows that pages visited (ie URL) can be determined quite effectively.

Share Improve this answer

answered Aug 14, 2015 at 16:03

Follow




pbhj

298 ● 1 ● 3 ● 16

That would really only be feasible on very small sites, and in those cases, the theme/tone/nature of the site would probably still be about the same on each page. – [Cameron](#)
Aug 21, 2015 at 18:39

- 5 From the citation I gave: "We present a traffic analysis attack against over 6000 webpages spanning the HTTPS deployments of 10 widely used, industry-leading websites in areas such as healthcare, finance, legal services and streaming video. Our attack identifies individual pages in the same website with 89% accuracy [...]". It seems your

conclusion as to feasibility is wrong. – [pbhj](#) Sep 4, 2015 at 10:15

- 3 For anyone interesting in reading more about this sort of vulnerability, these types of attacks are generally referred to as [side-channel attacks](#). – [Dan Bechard](#) Apr 18, 2016 at 18:18 
-



9



You can not always count on privacy of the full URL either. For instance, as is sometimes the case on enterprise networks, supplied devices like your company PC are configured with an extra "trusted" root certificate so that your browser can quietly trust a proxy (man-in-the-middle) inspection of https traffic. This means that the full URL is exposed for inspection. This is usually saved to a log.

Furthermore, your passwords are also exposed and probably logged and this is another reason to use [one time passwords](#) or to change your passwords frequently.

Finally, the request and response content is also exposed if not otherwise encrypted.

One example of the inspection setup is described by [Checkpoint here](#). An old style "internet café" using supplied PC's may also be set up this way.

Share Improve this answer

Follow

answered Nov 17, 2016 at 2:49



[Don Gillis](#)

131 ● 1 ● 2



8



Although there are some good answers already here, most of them are focusing in browser navigation. I'm writing this in 2018 and probably someone wants to know about the security of mobile apps.

For mobile apps, if you control both ends of the application (server and app), as long as you use HTTPS **you're secure**. iOS or Android will verify the certificate and mitigate possible MiM attacks (that would be the only weak point in all this). You can send sensitive data through HTTPS connections that it will be encrypted **during transport**. Just your app and the server will know any parameters sent through https.

The only "maybe" here would be if client or server are infected with malicious software that can see the data before it is wrapped in https. But if someone is infected with this kind of software, they will have access to the data, no matter what you use to transport it.

Share Improve this answer

answered Aug 3, 2018 at 0:08

Follow



Ricardo Gonçalves

5,064 ● 3 ● 22 ● 30



8



While you already have very good answers, I really like the explanation on this website:

<https://https.cio.gov/faq/#what-information-does-https-protect>

in short: using HTTPS hides:



- HTTP method
- query params
- POST body (if present)
- Request headers (cookies included)
- Status code



Share Improve this answer

answered Apr 5, 2020 at 13:37

Follow



[pedrorijo91](#)

7,845 ● 9 ● 46 ● 86



7



Linking to my answer on a [duplicate question](#). Not only is the URL available in the browsers history, the server side logs but it's also sent as the HTTP Referer header which if you use third party content, exposes the URL to sources outside your control.



Share Improve this answer

edited May 23, 2017 at 12:10



Follow



Community Bot

1 ● 1

answered Apr 15, 2016 at 15:28



[JoshBerke](#)

67k ● 25 ● 129 ● 170

Providing your third party calls are HTTPS aswell though this isn't an issue right? – [Liam](#) Nov 10, 2016 at 16:46 ✎

3 It'd be encrypted with the third parties certificate so they could see the URL – [JoshBerke](#) Nov 11, 2016 at 18:03



1



Additionally, if you're building a ReSTful API, browser leakage and http referer issues are mostly mitigated as the client may not be a browser and you may not have people clicking links.

If this is the case I'd recommend OAuth2 login to obtain a bearer token. In which case the only sensitive data would be the initial credentials...which should probably be in a post request anyway

Share Improve this answer

Follow

answered Aug 22, 2018 at 8:03



Chris Rutledge

357 ● 3 ● 7



Highly active question. Earn 10 reputation (not counting the [association bonus](#)) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.