All.Net Analyst Report and Newsletter

Welcome to our Analyst Report and Newsletter

Encrypted is not Secured, and https is even worse

People who think they know believe the messaging from browser providers who say things like “Site Not Secured” when accessing a site that runs hypertext transfer protocol (http) instead of http ‘secure’ (https). Let’s stop this myth today.

What is https supposed to do?

Sorry for the introductory nature of this article, however, http version 1.1[1] of 1997 was obsoleted by a new version of 1.1[2] obsoleted by RFC’s 7230, 7231, 7232, 7233, 7234, 7235 and eventually went on the standards track. The “The Secure HyperText Transfer Protocol”[3] (S-HTTP) was the 1999 start of trying to provide “security services for transaction confidentiality, authenticity/integrity and non-repudiability of origin.” However, the one that lasted was “The Transport Layer Security (TLS) Protocol” which “The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery.”[4]

Let’s assume it actually does that…

To be clear, I am not agreeing that these objectives are actually accomplished by https, but rather assuming for the moment that these things are true.

Note first the phrasing:

- **allows** – to be specific, does not force this to be the case
  - For example, https can run with encryption that is trivial and can be forced into trivial encryption by machine in the middle attack or server-side or proxy settings.
  - And even if you start with https, other protocols are almost always involved that do not necessarily provide even these limited design features.
- **client/server applications** – only applications – only client/server
  - For example, even if because the ‘front end’ server or ‘proxy’ server connected to by your browser provides these limited design features, does not mean that they hold true end to end, which they almost never actually do. Consider that your https connection leads to a server that looks things up in a database somewhere and you have no idea of what that communication uses.
  - Also note that it only encrypts at the https protocol level, not at the lower Internet Protocol (IP) level. Among the implications are the demonstrated fact that typing things like passwords which go one character at a time are subject to timing attacks, and other similar traffic analysis is commonly effective at gaining access to the content you put in and the server sends back.

[1] https://www.rfc-editor.org/rfc/rfc2068.txt
[2] https://www.rfc-editor.org/rfc/rfc2616.txt
[3] https://www.rfc-editor.org/rfc/rfc2660.txt
[4] https://www.rfc-editor.org/rfc/rfc5246.txt

- **to communicate** – only communications and not necessarily all of them
    - Only the communications between the endpoints of the encryption are actually covered in any way. And of course there are communications used to setup the encrypted communications, and some of them are not encrypted.
    - And of course just because the communications are encrypted does not mean the other components of activities in the server are also encrypted or otherwise protected. For example, if you send a file over https, and it is stored on a server, your use of https does not mean that the file is not stored in plaintext in an Internet-accessible server that has no effective protection.
- **designed to prevent** – may not actually prevent – and only prevent
    - Only prevention is designed into it. Not deterrence, not detection and response, not anything else. Note for example that encrypted content is intended to make it very hard to see the content, which means that detection between the end points based on the content being sent is far harder, and in many cases, infeasible. Which means that it doesn't prevent, but in fact prevents detection of malicious content transmitted either way.
    - And then there is the question of whether it actually prevents any of the three...
- **eavesdropping** – observing – sort of – confidentiality
    - Because the protocol does not deal with traffic analysis, all of the methods of traffic analysis are available to attackers. This includes knowing each site you went to, how many pages you viewed and their size and speed characteristics, how long you spent there, and most of all the other things you get from Web server analytics on usage. If I can decode this, I can likely also get to your search terms and similar things.
    - But who are we kidding? Applications you use over the Internet often nowadays record your keystrokes and transmit them back to the companies you communicate with, or the advertising forms or other providers involved in the totality of content they provide. So everything you type in and everything you get back is observable by other means and often observed and recorded anyway. On the other hand, you cannot tell what they are doing because much of their communication is encrypted – within the SSL tunnel by their add-on methods. They don't necessarily trust your encryption after all...
- **tampering** – alterations – integrity of content
    - Of course we have to start with the fact of garbage in garbage out. Much of the content over the Internet is garbage anyway in terms of integrity, so preventing altering it in transit doesn't do anything to fix that problem. And of course to do mass tampering, it's better to do it on the server anyway. Unless they are targeting just you for the tampering. In which case they also have your endpoint and lots of other places along the way. And of course if they get copies of the keys, they can tamper undetected, which means you will believe it is not tampered because of the protocol whereas you might worry about it with only using http. So a false sense of security is what you get.

    - Did I mention that the keys to encryption and the entire process depends on a multitude of other servers not involved in http communications? Yes, that's right, there are many more places an attacker can tamper in an SSL (https) encrypted session than in an http (not encrypted) communication. So it might actually be that there is more opportunity for tampering with https
- **message forgery** – only messages – forgery is technically a criminal violation
    - The term forgery is actually a term of the law, and does not apply more generally to false attribution, which is what I think they were trying to convey.
    - Of course it is only intended to cover message 'forgery', and not things like site 'forgery'. So at best it only makes certain that the fake messages sent by the fake server are properly attributed to the server being faked and not necessarily the faker.

What does https actually do?

As it turns out, https does some very important things you might want to understand:

- **More dependency → more failure modes:** It increases the dependency on 3rd parties and additional servers, connections, and content. That means there are more places for more people to cause less easily detected **eavesdropping**, **tampering**, **message 'forgery'**, and other bad things.
- **Worse performance → more time and energy:** It increases the time it takes to get answers and perform tasks, and along the way consumes more energy by orders of magnitude than were consumed without it.
- **False sense of security:** It clearly not only gives users a false sense of security, but also causes them to have an opposite sense of insecurity for http sites.

It also provides relatively transparent (to the user) encryption of messages between endpoints that makes it harder and less convenient/more expensive for a party with access to the intervening infrastructure to gain access to the content, execute machine in the middle attacks, and create false content offered as true.

What does https NOT do?

And then there are the many things https has no effect on, including the systems at the endpoints and their content, the systems they connect to and use to deliver their value, the other components of successful Internet communications like DNS and lower-level protocols, and the content actually transmitted and its potential for harm. Then there are the broad categories of availability (negative effect), use control, accountability (negative effect), transparency (negative effect), and custody (possible slight positive effect). Or in other words, everything else you might associate with the term "secure".

Conclusions

https does not make anything 'secure', and in fact brings about a false sense of security. It is portrayed for consumers as a panacea which in fact it is possibly even harmful in many cases. That's not to say we shouldn't use it, but it is to say we should not keep treating it as universally appropriate. It us meaningful and useful for certain things, and potentially harmful for other things. Keep it in perspective.