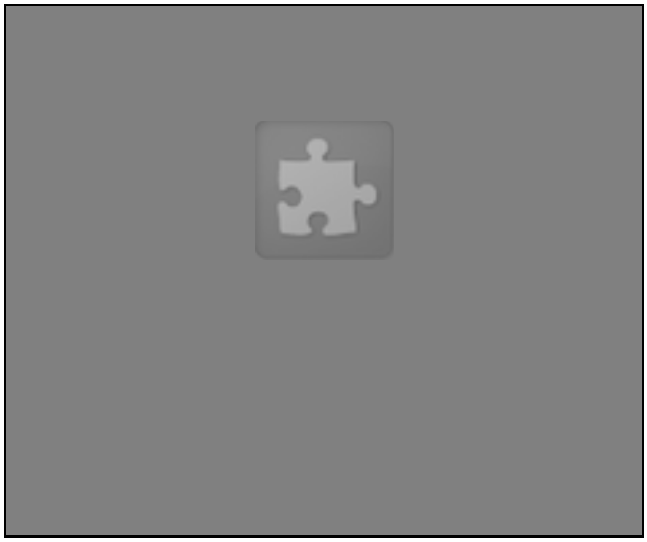
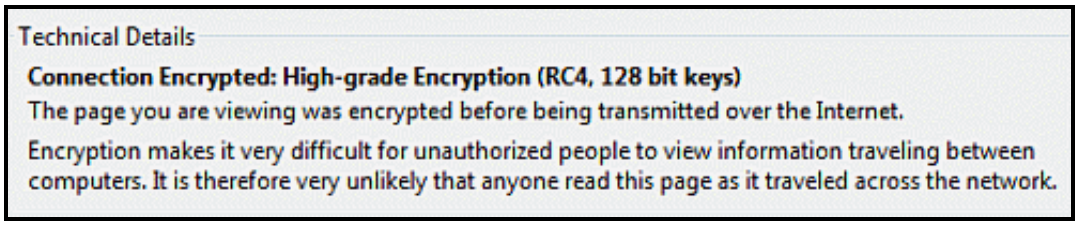
Published on *Computerworld Blogs* (http://blogs.computerworld.com)

Perfect Forward Secrecy can block the NSA from secure web pages, but no one uses it

By *Michael Horowitz*
Created *Jun 21 2013 - 4:49pm*

Suppose, for the sake of argument, that you wanted to spy on people using Microsoft's outlook.com website. First, you would need to capture requests to the site along with the returned web pages. But those pages are encrypted (sent via HTTPS rather than HTTP), so you would also have to break the encryption. Firefox tells us this is "very difficult" and "very unlikely" (see below).





But every lock has a key and outlook.com has a HUGE MASTER KEY. Anyone in possession of this master key can read the encrypted HTTPS pages. **All of them**. Every single encrypted web page that has ever been transmitted by outlook.com to millions of former Hotmail users can be decrypted with a single master key.

Of course, to read them all, you have to collect them all. This may not be a problem for the NSA.

It doesn't have to be this way. There is a server option called Perfect Forward Secrecy [1] that eliminates the single master decryption key.

But Microsoft doesn't use Perfect Forward Secrecy on the outlook.com servers. They are not alone. The vast majority of sites offering encrypted web pages, including *all* the big financial

institutions except one (detailed below), fail to employ Perfect Forward Secrecy.

Here I will explain how Perfect Forward Secrecy works, why most websites fail to employ it, how to detect when it is being used and point out two websites (one brutally popular) that do use it. Take a guess. What two companies do you think have their technical act together sufficiently to use such an obscure security feature?

It's a lot to cover, this blog is long and, at times, somewhat technical.

WIRETAPS

Multiple reports claim that the NSA taps directly into the communication backbone of the Internet. Edward Snowden, the NSA leaker, recently said [2] "We hack network backbones – like huge Internet routers, basically – that give us access to the communications of hundreds of thousands of computers without having to hack every single one."

Back in 2006, a former AT&T employee alleged that the NSA was tapping into the AT&T Worldnet service (see here [3] and here [4] and here [5]). CNET speculated back then whether it was even necessary [6] for the major telecommunication companies to co-operate with the NSA for them to install wiretaps.

As an analogy, think of this as if the NSA had cameras on the median of a highway, recording images of every vehicle as it passes.

One of the PowerPoint slides that Snowden leaked [7] showed that major Internet companies (Microsoft, Yahoo, Google, Facebook, PalTalk, YouTube, Skype, AOL, Apple) were assimilated into the PRISM [8] system over the space of a few years.

Even for the NSA collecting *all* the data traveling across the Internet on US soil is probably too big a job (at least until their new data center in Utah comes online).

Steve Gibson recently devoted an entire episode of his Security Now podcast [9] to this. He surmised, going back to the analogy, that the NSA was only monitoring some exits and entrances to the highway that is the Internet.

By its very definition, the Internet is a network of networks. Gibson's theory is that when the NSA wanted to spy on data coming and going from a particular company they put wiretaps at the junction point where that company's data centers are connected to the outside world (highway entrances and exits in the analogy).

The beauty of this scheme (from the point of view of the NSA) is that the spied upon company is

out of the loop, they would have no idea anything had happened.

Much of the data traveling over the Internet is un-encrypted leaving it easily accessible to a wiretapper.

But what about secure web pages, those transmitted via HTTPS rather than HTTP? HTTPS is purposely designed to foil just this sort of wiretapping.

Going back to the highway analogy, encrypted HTTPS web pages are the equivalent of a car with dark tinted windows. The NSA cameras on the median of the highway can't see into the car, but they know that someone is purposely hiding in there. Most likely, that's where the good stuff is.

It is said that the encryption used for HTTPS (provided by either the SSL or TLS protocols) is unbreakable, by and large. There have been some coding errors over the years, of course, and there were some weak encryption choices, but it has mostly held up to outside scrutiny.

Even if the NSA can't see past the tinted windows at the time of the recording, they would certainly keep the video in case a future technical breakthrough (i.e. quantum computing) would provide access. But there is no need to break through a wall, if the you have the key to the front door.

How can one piece of data decrypt every web page outlook.com has ever sent? Especially if the encryption is as good as experts claim. The explanation is a bit techie.

SHARED KEY EXCHANGE

Encrypted HTTPS connections (note that I have given up on the word "secure" here) start out with a dialog between your computing device and the web server computer hosting the site you want to visit securely. Below is a terribly inaccurate example of the handshaking conversation [10]that sets up the connection initially.

- **My computer:** hello www.joeswebsite.com I would like to have an encrypted conversation.
- **Server:** OK, here is my digital certificate and public key
- **My computer:** Nice certificate. Lets pick an encryption scheme, I support schemes 4, 9 and 12
- **Server:** Lets use encryption scheme 9 please
- **My computer:** OK, 9 it is. We need a shared encryption key (password) for this session. Since I'm in Boston, how about "LetsGoRedSawx"
- **Server:** Baseball is boring, but from now on, I agree to use "LetsGoRedSawx" as our shared password

The important point here is that **the end result of the setup conversation is a shared**

encryption key (think of it as a password). The technical term for this is symmetric encryption. If outlook.com sends you 99 web pages, each one was encrypted with the same key/password.

This is not, however, the problem since each computer uses a different shared encryption key/password when talking to joeswebsite.com. The problem has to do with the way my computer sends "LetGoRedSawx" (the shared encryption key) to the server. It does so encrypted with the servers public key, which lets the server decrypt it using its private key.

An explanation of public and private keys would make this long blog even longer. Suffice it to say the private key is something that only the web server running joeswebsite.com is supposed to know - hence the term "private". This private key is also what I was referring to in the beginning as the HUGE MASTER KEY.

Even though every computer picks a different shared encryption key, they are all sent to the webserver such that *whoever* knows the private key can decrypt not only the current session between my computer and joeswebsite.com, but *every session* between *every computer* and joeswebsite.com.

Suppose, for example, the NSA was recording all HTTPS encrypted traffic to/from joeswebsite.com in January. Then, in February, they learned the private key for joeswebsite.com. Almost always, that lets them decrypt everything from January, February, March and beyond.

It is as if every HTTPS session to joeswebsite.com were encrypted with the exact same password. That's not *actually* what happens, but in terms of spying on joeswebsite.com, it might as well be.

One key can unlock quite a gold mine of data.

PERFECT FORWARD SECRECY

The more secure way of handling things, as noted earlier, is called Perfect Forward Secrecy [11]. It uses a variation of the initial setup conversation where the server chooses a temporary encryption key. The point of this, as described by Vincent Bernat [12] is to keep "todays information ... secret even if the private key is compromised in the future."

With Perfect Forward Secrecy, anyone possessing the private key and a wiretap of Internet activity can decrypt *nothing*.

The standard bearer for Perfect Forward Secrecy is Google, which converted to it late in 2011. It was announced in a blog by Adam Langley [13]

> Most major sites supporting HTTPS operate in a non-forward secret fashion, which runs the risk of retrospective decryption. In other words, an encrypted, unreadable email could be recorded while being delivered to your computer today. In ten years time, when computers are much faster, an adversary could break the server private key and retrospectively decrypt today's email traffic ... An adversary that breaks a single key will no longer be able to decrypt months' worth of connections; in fact, not even the server operator will be able to retroactively decrypt HTTPS sessions.

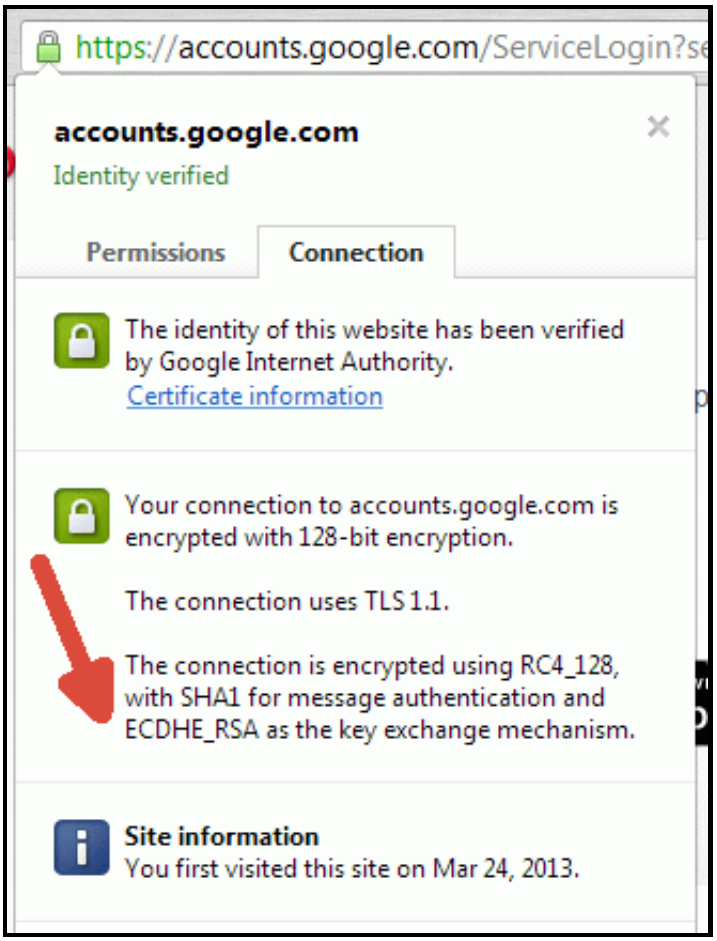
Langley points out that Google's Chrome browser can be used to verify that Perfect Forward Secrecy is in place. For Chrome on Android 4.1 and iOS 6, click on the green padlock icon, then look for the key exchange mechanism. For Chrome on Windows and OS X, after clicking on the green padlock, you need to click on the Connection tab before you can see the type of key exchange.

A shared key exchange using **ECDHE_RSA** is good (forward secret), one using **RSA** is bad.

Frankly, this is way over my head. That said, ECDHE_RSA stands for [14] elliptic curve, ephemeral Diffie-Hellman, signed by an RSA key. Say that three times fast.

The critical letter is the second "E" for ephemeral, which in turn, means that a temporary encryption key was used.

The only other *good* key exchange is the **DHE_RSA** employed at bloomberg.com (again, the "E" is for ephemeral).



Perfect Forward Secrecy and ephemeral Diffie-Hellman key exchanges are fairly obscure stuff. Until a few days ago, I had never heard of either. It's so obscure that, except for Chrome, web browsers don't externalize it. On Windows 7, I checked Firefox 21, Internet Explorer 9, Internet Explorer 10 and Opera 12.15. None said anything about the key exchange. Even the SSL server test [15] from SSL Labs doesn't spell it out clearly.

This ends the hard part.

PERFECT FORWARD SECRECY IN THE WILD

Most websites run without Perfect Forward Secrecy.

When you login to Amazon.com, for example, it's not used. Same for yahoo.com, americanexpress.com, chase.com, wellsfargo.com, citibank.com, bankofamerica.com, capitalone.com, bnymellon.com, citizensbankonline.com, fidelity.com, vanguard.com, oppenheimerfunds.com, etrade.com, scottrade.com, schwab.com, jpmorgan.com, macys.com, llbean.com, bloomingdales.com, nordstrom.com, walmart.com, target.com, homedepot.com, newegg.com, apple.com, lenovo.com, dell.com, hp.com and paypal.com.

But, when you login to gmail.com there is forward secrecy, at least with Google's Chrome and other modern* web browsers.

If Perfect Forward Secrecy offers extra security, why do so few websites use it?

Perhaps some website operators aren't aware of it. I doubt, however, that applies to Paypal, American Express and the banks for whom security is critical. Conspiracy theorists might suspect that some organizations were compelled not to use it. Nikos Mavrogiannopoulos, a mathematician with a PhD in cryptography, said that most servers prohibit the use of Perfect Forward Secrecy [16].

Chances are, that the answer is simply because it takes more computing horsepower.

Mavrogiannopoulos measured the overhead of Perfect Forward Secrecy. Without it, his test computer could perform 460 transactions per second (the higher the number the better the performance). Using the ECDHE-RSA that Google employs, he got 352 transactions/second, a 23% reduction. With the DHE-RSA used by Bloomberg, he got only 98 transactions/second.

Vincent Bernat found [12] the overhead to be between 15% and 27% using ECDHE-RSA. However, that was for "full" TLS handshakes. Apparently the handshakes can also be resumed, something that's beyond my understanding.

WHO HOLDS THE KEYS?

Trivially easy, global spying on encrypted (what others call "*secure*") web pages is only possible if the NSA can get the private keys of the companies they want to spy on.

Can they?

At the most basic level a private key is just a file and files can be copied without a trace. So, it's not hard to imagine a number of possible scenarios by which the private key might leak out.

For one, the computer running the website could be hacked either remotely, or if some breaking and entering is involved, locally. Then too, computers are administered by people and if the right person is made an offer they can't refuse ...

If nothing else, the recent Edward Snowden revelations remind us that the government has compelled companies to turn over information while forbidding them from saying anything publicly.

So, the next time someone near you enters sensitive information into a web page ask if the site employs Perfect Forward Secrecy. Tell them that without it, a single piece of data is all that prevents the NSA from passively spying on every user of that website in a totally undetectable way. If you read this far, you deserve to gloat.

Finally, a thank you to Google and Bloomberg for making your customers safer even though none of them know about it.

---

*Perfect forward secrecy at Gmail also works with modern versions of IE and Firefox. Google said it does not work, however, with IE on Windows XP. As for other browsers, Google said that their servers use forward security with any client that offers ECDHE. Does your browser support ECDHE? See Cipher Suites Supported by Your Browser [17].

UPDATE: June 25, 2013. Over at SSL Labs, they just today added a new feature (still in the experimental phase) that displays information about the use of Perfect Forward Secrecy [18] by a website.

UPDATE: June 26, 2013. Yesterday was a big day for Perfect Forward Secrecy. Netcraft also did a big writeup on it. See SSL: Intercepted today, decrypted tomorrow [19].

UPDATE: June 26, 2013. PFS seems to be gaining steam. The Register did a story on it today: A simple SSL tweak could protect you from GCHQ/NSA snooping [20]. Not worth reading though, it is just a re-hash of the yesterdays blog at SSL Labs.

data privacy encryption government spying HTTPS NSA PRISM SSL Encryption Network Security

Source URL: http://blogs.computerworld.com/encryption/22366/can-nsa-see-through-encrypted-web-pages-maybe-so

Links:
[1] http://en.wikipedia.org/wiki/Perfect_forward_secrecy
[2] http://www.guardian.co.uk/world/2013/jun/09/edward-snowden-nsa-whistleblower-surveillance
[3] https://www.eff.org/files/filenode/att/presskit/ATT_onepager.pdf
[4] http://arstechnica.com/uncategorized/2006/04/6585-2/
[5] http://en.wikipedia.org/wiki/Room_641A
[6] http://news.cnet.com/NSA-eavesdropping-How-it-might-work/2100-1028_3-6035910.html
[7] http://www.wired.com/threatlevel/2013/06/snowden-powerpoint/#slideid-57994
[8] http://www.guardian.co.uk/world/2013/jun/06/us-tech-giants-nsa-data
[9] https://www.grc.com/sn/sn-408.htm
[10] http://en.wikipedia.org/wiki/Transport_Layer_Security
[11] http://en.wikipedia.org/wiki/Transport_Layer_Security#Perfect_forward_secrecy
[12] http://vincent.bernat.im/en/blog/2011-ssl-perfect-forward-secrecy.html
[13] http://googleonlinesecurity.blogspot.com/2011/11/protecting-data-for-long-term-with.html
[14] http://www.imperialviolet.org/2011/11/22/forwardsecret.html
[15] https://www.ssllabs.com/ssltest/index.html
[16] http://nmav.gnutls.org/2011/12/price-to-pay-for-perfect-forward.html
[17] https://cc.dcsec.uni-hannover.de/
[18] https://community.qualys.com/blogs/securitylabs/2013/06/25/ssl-labs-deploying-forward-secrecy
[19] http://news.netcraft.com/archives/2013/06/25/ssl-intercepted-today-decrypted-tomorrow.html
[20] http://www.theregister.co.uk/2013/06/26/ssl_forward_secrecy/