Secure Browsing Guide

Authors: @pseudophed, @scumbagjames

Crypt.To: https://cryptoparty.org/wiki/Toronto

Follow Us On Twitter!: @CryptToronto

Introduction

Cryp.TO is a nonpartisan organization who's purpose is to obtain acceptance of the daily use of encryption by everybody. New pursuits in web research have propelled technologies that make the task of tracking a person online trivial. Reports of networks consisting of corporations and advertisement agencies being given the ability to passively monitor information to assist in building portfolios for future sales and/or potential for harassment, constructing portfolios of behavior based on how you search Google or what you buy on Amazon. This is an issue that must be addressed.

Overview

This document was written by members of Cryp.TO to assist the average end user (AEU) in learning the importance of good browsing habits and strong browser hardening. This document is also just a guide and by no means the most comprehensible guide on the topic. Many guides exist on the internet and should be researched accordingly. However, we do believe that this is a great starting point and have attempted to cut out the elements we believe are stopping encryption from being accepted.

Security begins from the local policy and continues on through to browsing policy.

Behavior Tracking

"New web technology has created many unexpected ways for corporations to track your web activity without your knowledge. Countless advertising networks are able to secretly monitor you across multiple websites and build detailed profiles of your behavior and interests.

New threats include "super-cookies" like Adobe's "Local Shared Objects" and Microsoft's "User Data Persistence." They include semi-legal data-sharing agreements between Internet service providers and data warehouses like Phorm and NebuAd. And they include social networking websites that allow advertisers too much access to their users' behavior and data."

https://www.eff.org/issues/online-behavioral-tracking

Events Involving Behavior Tracking:

NDP Immigration Minister Emailing LGBT Canadians

Information was farmed from an on-line petition in which the user generated a template-body for an email. The office of the recipient farmed the email address from the emails and then used it as an on-line mailing list to spam these harvested email addresses.

http://www.cbc.ca/news/politics/story/2012/09/24/iran-refugee-gay-lesbian-email.html

Target, Targets the Expecting

In this case, Target was utilizing a algorithm based on products a customer purchases to determine their "pregnancy score". This algorithm could also then predict a due date and would send coupons to the customer targetted for each stage of pregnancy.

http://www.forbes.com/sites/kashmirhill/2012/02/16/how-target-figured-out-a-teen-girl-was-pregnant-before-her-father-did/

Human Resources Department is now Datamining

HR agencies and departments have been reported to generate numbers based on the productive output of employees using algorithms to predict who will add the highest value to a company over their lifetime. Not only this, they are also purported to "predict leaders" and which potential employees may be the most prone to accidents. This is digitally gleaned by snooping on unencrypted emails and by noting the traffic sent to and from certain accounts.

http://www.businessweek.com/stories/2009-03-11/data-mining-moves-to-human-resources

Definitions

Cookies

A cookie, also known as an HTTP cookie, web cookie, or browser cookie, is usually a small piece of data sent from a website and stored in a user's web browser while a user is browsing a website. When the user browses the same website in the future, the data stored in the cookie can be retrieved by the website to notify the website **of the user's previous activity.**

Session cookie

Session cookies are pieces of data that exist to log a user in and out of a system. A user's session cookie for a website generally only exists while the user is reading and navigating the website. Web browsers by default delete session cookies when the user closes the browser.

Persistent cookie

A persistent cookie will outlast user sessions. If a persistent cookie has its Max-Age set to 1 year, then, within the year, the initial value set in that cookie would be sent back to the server every time the user visited the server. This could be used to record a vital piece of information such as how the user initially came to this website. For this reason persistent cookies are also called tracking cookies.

Secure cookie

A secure cookie has the secure attribute enabled and is only used via HTTPS, ensuring that the cookie is always encrypted when transmitting from client to server. This makes the cookie less likely to be exposed to cookie theft via eavesdropping.

Third-party cookie

First-party cookies are cookies set with the same domain (or its subdomain) in your browser's address bar. Third-party cookies are cookies being set with different domains from the one shown on the address bar (i.e. the web pages on that domain may feature content from a third-party domain - e.g. an advertisement run by www.advexample.com showing advert banners). For example: Suppose a user visits www.example1.com, which sets a cookie with the domain ad.foxytracking.com. When the user later visits www.example2.com, another cookie is set with the domain ad.foxytracking.com. Eventually, both of these cookies will be sent to the advertiser

when loading their ads or visiting their website. The advertiser can then use these cookies to build up a browsing history of the user across all the websites this advertiser has footprints on.

Zombie cookie

Some cookies are automatically recreated after a user has deleted them; these are called zombie cookies. This is accomplished by a script storing the content of the cookie in some other locations, such as the local storage available to Flash content, HTML5 storages and other client side mechanisms, and then recreating the cookie from backup stores when the cookie's absence is detected.

Blacklist/Whitelist

The idea behind blacklist/whitelist is a simple DENY/ALLOW security model.

Whitelist = ALLOW

Blacklist = DENY

A good example of this is in corporate proxy servers. For example, have you noticed that when you are at your desk at work, you can't access Facebook.com? Most corporate proxy's "Blacklist" social media sites (YouTube, Facebook, Twitter, etc.) as a way of controlling corporate resources and making you more productive. Under the same idea, company intranet sites, research sites, etc. are "Whitelisted".

DNS

Domain Name Service. This is the underlying infrastructure which allows you to enter logical names, and map it to IP addressing. It is what enables you to type in www.google.com into a web browser, and it directs you to the correct server. DNS is like house addresses. Instead of saying "go down 2 streets, turn right and the 5th house on the left", it is logical to say "123 maple street". DNS operates under the same rules.

HTTPS

HTTPS is the protocol HTTP with Secure Socket Layer attached to it, making it more secure and harder for negative operaters to fool people utilizing fake sites (for malware downloads, etc). HTTPS is useful in two ways - for one, it encrypts the stream of transmission from sender to reciever. Second and one that is often overlooked, HTTPS also verifies site's integrity by utilizing certifcates and certificate authorities chain of trust. If a certificate authority has not verified a websites e-commerce site, you may want to look at the untrusted certificate closely and determine if the website whom they say they are.

Hash

A hash function is any algorithm or subroutine that maps large data sets of variable length, called keys, to smaller data sets of a fixed length. For example, a person's name, having a variable length, could be hashed to a single integer. The values returned by a hash function are called hash values, hash codes, hash sums, checksums or simply hashes.

In layman terms, a "hash" is a way of masking a "message". For example, an Atbash Cipher is a rudimentary hash.

Example:

LOL could be MPM, where the hash would be:

n = n + 1

MD5 hashing

MD5 is a hash function that is commonly used to check data integrity. In relation to browsing security MD5 hashing is used to verify that the file you've downloaded is the correct one and not something that has been modified from its original version. Files can be distributed via different mediums and thus the signature acts as a central authority on whom the file belonged to, as well as if the file is still in the same state (unchanged) since it was published for that download.

MD5 hashing is by no means secure and should not be relied on for important authentication methods (SSL certificates, CAC signatures).

Malware

Malware can be seen as small applications or scripts which run and hamper computer systems. Malware differ from viruses because they are not typically as openly malicious (destroying data, outputting a message etc). Typically, malware is more inclined to redirect browser requests to infected websites, logs sites visited or do other 'background' attacks.

Virtualization

Virtualization is defined as 'the act of virtualizing'. Virtualizing is then defined as 'to run a program in virtual storage' or 'to simulate some effect or condition on a computer'. When you are virtualizing, you are essentially making storage space (MB, GB's) out of memory. One way to further understand this logic looking at the transformation of water to ice - you do not ever have 'more memory' (water) but can take this memory and turn it into another object (ice): an application or an entire operating system.

Virtualization is extremely important to security policy. When an operating system is virtualized, for example, the virtual machine separates the host (operating system running the virtualizing software) from the guest (the operating system inside the virtualized environment) and severely limits any further exploitation of the Host.

Real world examples of where this would come into play:

"Bob" downloads an infected file without virtualizing his operating system and gets his entire Windows ME installation corrupted. "Doug" the ever vigilant and paranoid of the McKenzies, virtualizes his operating system and browses through it. Even though he's updated his Firefox to the most recent branch, he gets hit by an unexpected Java exploit (that wasn't gonna happen anyway, right?) Luckily, his Windows 7 was running inside a virtual machine he made. After noticing that his virtual machine was compromised, all he has to do is close the machine and create another one.

Sources:

1. 'Does the virtualization of a web browser prevent a virus infection'?

http://www.symantec.com/connect/blogs/does-virtualization-web-browser-prevent-virus-infectionyou

2. 'Virtualize your browser to prevent drive-by malware attacks'

https://www.networkworld.com/newsletters/techexec/2010/090610bestpractices.html

Tools:

Spoon: "Run any browser from the web in an isolated virtual environment"

https://spoon.net/browsers

Drive-by Downloading

This term is used to describe two situations. The first, a scenario in which a download occurs where the user is unaware the download/execution is occurring. The second is a scenario when a potentially infected file is downloaded *without* the consequences being fully realized.

Password Management

This evaluation has been submitted by (@scumbagjames)

KeePass vs. 1Pass vs. Lastpass

Pros/Cons are based off comparision to other password management techniques

KeePass -- keepass.info

Pros:

- KeeFox Plugin for ease of use
- Sync with Android/iPhone
- Can import over 30 other password managers
- Open Source

Cons:

- Does not automatically fill forms for ease of use
- Many configuration settings may break down ease of use
- Interface isn't the most appealing
- Requires .NET download

LastPass -- *lastpass.com*

Pros:

- Available as long as connection is available
- Plugins and Betas have short lifecycle (fast releases)
- Generate secure notes
- Generate strong keys that are stored regardless if copied correctly
- Stored remotely but still encrypted upon transmission/receiving
- Importing option for most password managers in FF/Chrome
- Even LastPass employees cannot access the remotely stored password [2]

- Anti-keylogging virtual keyboard

Cons:

- Remote storing opens up possibility of unrelated compromise
- Remote storing also does not solidify permanent privacy policy [1]
- Mobile option is only available at a charge
- Charge is subscription
- Only accessible through browser
- Closed source

1Pass

Pros:

- Implementation with DropBox for ease of use
- Can be stored locally
- Interface is an application rather than a browser extension
- Wallet extension is easy to use
- Tagging of information for placement of pictures alongside data
- PBKDF2

Cons:

- Costs money to utilize, no free version.
- Chrome extension is invasive and not easy to use
- No two factor authentication built in
- Remote storing functionality is related to Dropbox security rather than it's own
- Remote functionality is dependant solely on Dropbox, raises accountability questions

Conclusion

KeePass isn't as good for the remote storing option as LastPass and the interface is sub-standard while compared to either LastPass's or 1Pass's. Thus my conclusion is that the actual

debate is up to 1Pass vs LastPass. I personally find it to be advantageous to utilize remote storing of files and passwords and thus I've based my decision solely off how each service remotely stores it's data. Understandably, 1Pass's locally storing option is obviously more reliable (as the only chain of trust to be broken is your own security policy) and trustworthy than a remote blob of data on a server that is out of your control. However, the host of features, ease of use and actual data integrity of LastPass explains why they take the vote from me. If you're more inclined to only trust your local policy (which is understandable) then there is no choice - 1Pass takes the cake.

Discovering that 1Pass relies on Dropbox primarily for remote storing is extremely worrying and should bother even the most mediocore of the paranoid amongst us. Dropbox is notorious for having it's services compromised and this shouldn't surprise anyone! Dropbox is a storing mechanism first, security device after. [4]. Therefore the Dropbox/1Pass combination raises questions about accountability held by either services - Dropbox is a storing mechanism and thus will not have security first in mind. LastPass is a service that recieves profit solely on it's success as a security mechanism, I find the interface to be intuitive as well as powerful which is what CryptoParty strives to achieve to provide.

Resources:

[1] http://www.techerator.com/2011/03/why-i-left-lastpass-for-1password/
[2] http://lastpass.com/whylastpass_technology.php
[3] http://www.pcmag.com/article2/0,2817,2408063,00.asp

[4] http://dereknewton.com/2011/04/dropbox-authentication-static-host-ids/

Other browser/online password managers:

- Roboform (can utilize fingerprint authentication)
- Dashlane

Browser Management

Browser Choice:

Chrome vs. Firefox vs. Internet Explorer vs. Safari

Chrome Pros:

- Clean and light UI
- Good browsing experience – good compatibility and speed

Chrome Cons:

- Every keystroke in address bar is time-stamped and sent to Google cloud http://www.zdnet.com/news/google-bows-to-keystroke-privacy-concerns/220444

Firefox Pros:

- Great security extensions
- Great overall security experience. Native pop-up blocker stops nearly all pop-ups and general adware

Firefox Cons:

- UI is ugly, and browsing experience isn't as good as other browsers
- Memory usage. Firefox tends to consume a lot of memory, which can cause it to crash when a lot of tabs are open concurrently
- Downloads cannot be resumed

Internet Explorer Pros:

- Arguably the best compatibility and browsing experience
- Built into Windows by default
- Updates regularly from Microsoft. Easy to update as part of regular Windows Updates

Internet Explorer Cons:

- Slow
- IE is targeted heavily by malware, adware and exploits
- Poor security. Runs Active X and Java code with very little protection or prompting under default settings
- No extensions

Safari Pros:

- Built into OSX by default
- Stable, good browsing experience

Safari Cons:

- Not updateable through any regular means
- No extensions
- Mediocre security

Mobile Browsing

Firefox:

"Firefox for Android uses best practices for security testing and adheres to Mozilla secure-development guidelines just like desktop Firefox. Security testing called fuzzing is used, to make sure Firefox for Android is robust enough to handle all kinds of crazy data without crashing. We do specific testing for the ARM processor, conduct thorough design reviews, code reviews and perform hostile testing in the same form as is done for desktop Firefox."

http://support.mozilla.org/en-US/kb/how-does-firefox-mobile-android-provide-secure-mob

Firefox on the mobile development angle, has all the same security features that make Firefox a great desktop browser. It has automated updates, so there is a continuous feedback of bug fixes which address new exploits. It also has a great encrypted password utility which connects to your desktop password manager (within Firefox). This means you won't need to type sensitive passwords into mobile devices where people can be watching and grab your password.

Chrome:

"A central design point of the Android security architecture is that no application, by default, has permission to perform any operations that would adversely impact other applications, the operating system or the user. This includes reading or writing the user's private data (such as contacts or e-mails), reading or writing another application's files, performing network access, keeping the device awake, etc."

https://www.google.com/intl/en/chrome/browser/features.html#security

Chrome supports updating through the Play Store or iTunes. One thing to note, is that Chrome currently does not support 'SafeBrowsing' on the Android platform. This is not great, but not necessarily a deal-breaker. SSL is still fully supported, so it helps to squelch wireless snoops.

Safari:

Safari does not natively support automated updating, which means the user needs to already be aware of any security patches they need to update it with, if the patch is available at all. Typically it just gets patched when the next iOS revision is released. Safari does support disabling javascript, etc., however this is buried within the menus and is not easy to disable quickly.

What are certificates?

Certificates are a file which is stored in your computer "keystore". A "keystore" in this instance, is a vault for your computer to put trusted things in.

The certificate will contain the following information:

- the domain name which the certificate has authority for
- the name of the issuer
- what issued the certificate

How to verify certificates? (to a reasonable certainty)

There are two critical fields in the certificate which we can use to deduce the authenticity of the certificate:

Issued By:
Issued To:

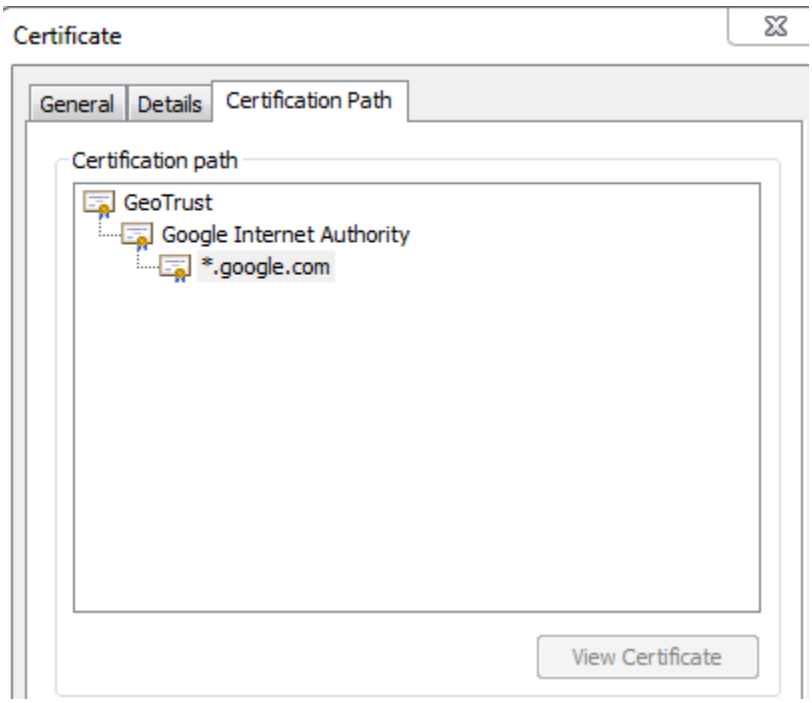
First, we look at the Issued By: field. This will contain the name of the issuing authority. Grabbing the name and performing a quick search should bring you to the registrar's page. Have a look around the site. If it seems sketchy, it probably is. There is a list of known, trusted registrar's, including (but not limited to):

- **GoDaddy**
- **Verisign**
- **Thawte**
- **Komodo**
- **GeoTrust**

Next, you want to look at the Issued To: field, and make sure it's issued to the domain you are accessing. If it's IssuedTo: *encrypted.google.com*, but the page you are viewing is encrypted.google.adware.com, that's a problem.

Here is an example of what the URL looks like when you are at an encrypted website:

Here are the certificate details:



In here we can see that this site's SSL certificate has been issued by GeoTrust, who is a very reliable CA.

Why and When it is Important

This is particularly important when you are doing sensitive things. Online banking, online shopping, email, etc. Anything where you wouldn't want the information you are sending to the website to be compromised.

It is particularly important to check the certificate yourself when you access a page with a certificate error. If this is an internal site (i.e. corporate intranet), when you look at the IssuedTo: field, the name will probably be different than the URL you are accessing. This is a typical mistake, and is not a malicious attack, typically. If you saw this same error on your bank's website, however, proceed no further.

Extensions: Ghostery, HTTPS Everywhere, NoScript, AdBlock

Ghostery:

Ghostery is a browser extension which is hooked into an adware database which tracks all of the adware providers, and gives you information on exactly what is being collected, traded, etc. It also gives you control to block everything, or block on a per-company basis (e.g. maybe you don't mind helping out google analytics, but want to block other adware companies).

HTTPS Everywhere:

HTTPS Everywhere is a browser extension that forces HTTP requests into HTTPS on pages where it is supported. It appears to also rewrite web requests which helps to stop mixed-mode pages from being displayed, and switching between secure and non-secure content.

NoScript:

Noscript is a browser plugin (limited to firefox, it appears) which disables java scripting, which in turn helps prevent you from being exploited. It also allows you to white-list certain webpages to allow scripting (such as bank, or trusted store, etc.)

AdBlock:

AdBlock is a browser extension which blocks advertisements from webpages. It does this seamlessly in the background so you're not even aware the ads are missing.

Cookie Management - Cookie Monster

Cookie Monster allows for easier managing of what sites a user allows to set cookies and what sites cannot.

By default, cookies are blocked, however a user can choose to allow cookies on a per-domain basis, which is useful if they want to allow cookies from a particular site (i.e. they trust it)

MD5 Hashing Verification

MD5 hashes can be viewed as a "file's fingerprint". The purpose of utilizing MD5 hashes is twofold: to verify that the file being downloaded is the originally intended file and verifying that the file has not changed since it was uploaded. To utilize this, all the author or uploader of the file has to do is use a MD5 tool (some listed below but that is by no means a comprehensive list) to list the hash near the file. On the flipside, when a user wishes to verify the file is still in the authors original posesssion and intact they can copy the hash from the page and verify in one of the many tools available. If the hash has changed by any means, the checksum will be

incorrect.

It is entirely possible that errors in data transmission can occur when a file is being downloaded so if the original checksum fails (and no damage or tampering on the file has been predicted) then downloading the file may sometimes fix an error. If the error keeps up, it is by all means a good idea to verify the file is where it is and owned by who it says it's supposed to be. For example: When a file is uploaded to a server, sometimes the accountability and verification of the servers hosting can be called to question (operated by fair admins, not been owned previously etc) and thus the data may be tampered with. This hash checking allows the internet to retain a sense of integrity in it's file transfers.

There are some web options available listed below for quick strings verification but every browser and operating system has tools available for MD5 hash verification. It is standard on most downloads to now have their associated MD5 checksum located beside the download link.

As for MacOS/Linux users, the application 'md5' via terminal is easy and quick to utilize.

Free MD5 Hash Calculator - http://md5calculator.chromefans.org/
WinMD5 Free - http://www.winmd5.com/
How To Check MD5 via Mac - http://osxdaily.com/2009/10/13/check-md5-hash-on-your-mac/
mdhashtool: Index http://mdhashtool.mozdev.org/

Two Factor Authentication

Two-factor authentication (TFA, T-FA or 2FA) is an approach to authentication which requires the presentation of two or more of the three authentication factors: a *knowledge* factor ("something the user *knows*"), a *possession* factor ("something the user *has*"), and an *inherence* factor ("something the user *is*").

Common Examples:

Password/PIN + RSA token

Everyone has seen someone with the little RSA tokens on the keychains. The user will log into a website, enter the token from the RSA key (possession factor), combined with a password or PIN (knowledge factor) and gain access.

Certificate + PIN

This is more common in the Enterprise world for VPN. A certificate will be issued to the laptop/computer which is "off-site". When the computer attempts to make the VPN connection, it will present the certificate (possession factor) and the user will enter a password (knowledge factor), and will gain access.

CAC + Password

CAC is used primarily in military/government facilities. CAC is a "Client Access Card" (otherwise known as a "Smart Card"), which contains a certificate. When needing to access something, the user places the CAC (possession factor) into a reader, and uses a password (knowledge factor) to complete the authentication, and gain access.

Email "Secret Questions/Security Verification Questions"

The secret is that email secret questions are actually horrible ways of verifying identity. With the use of Google and social media, information about people and their habits and their lives are in abundance and readily available for the pilfering.

Thus, using information that people also can find out about a user (i.e. birth date, phone number, etc.) is not a strong way to authenticate, or verify, a user's identity.

Common security practice is to use illogical answers for these questions.

Examples:

Q: "Your mother's maiden name?"

A: bluemountain

Q: "Your phone number?"

A: 2011010101