Mobile privacy: Tor on the iPhone and other unusual devices

Marco Bonetti
<mbonetti@cutaway.it>

May 2, 2010

Abstract

Tor is a software project that helps you defend against traffic analysis, a form of network surveillance that threatens personal freedom and privacy, confidential business activities and relationships, and state security. Tor protects you by bouncing your communications around a distributed network of relays run by volunteers all around the world: it prevents somebody watching your Internet connection from learning what sites you visit, and it prevents the sites you visit from learning your physical location.

Unfortunately, with the new features of HTML5 and browser built-in geolocation being pushed into the Web2.0 world and on mobile phones and browser, it's becoming harder and harder to keep the users' privacy safe. This paper will describe the problems which are arising around the use of these new technologies and how they can be (ab)used to attack Tor users. It will also describe where the development is going to protect mobile phone users privacy and let them survive their own devices.

1 Introduction

Tor is becoming more and more popular, it's no surprise that TorProject.org launched this year the "Help us reach 5000 relay in 2010!" drive [1] [2]. Unfortunately, such growth is not always followed by adoption of secure browsing behavior and new privacy exploit techniques are always in the development. If this is not enough, with the rapid growth and diffusion of mobile devices, it's becoming really difficult for the end user to protect his very own privacy.

Section 2 will describe the current scenario about secure communication available for mobile phones. Section 3 will talk about the current availability of Tor clients for mobile phones and the following chapters will go into details: Section 4 will describe the working port for the Chumby One multimedia alarm clock, Section 5 will talk about the Nokia N900 port, Section 6 will introduce the Android port and, finally, Section 7 will describe my work in porting Tor for the iPhone platform, how it can improve mobile privacy communications and which problems arise when using it on such platform. Finally, Section 8 will talk about what can be done next.

2 Mobile Phones (In)Security

The topic of mobile phones security saw its born in 2008 with the first conferences on the subject and literally exploded in 2009. The reasons behind this massive growth rely on five factors: phones are considered as something personal, we bring them everywhere we go. Phones are critical devices: they collect phone call logs, email and SMS; the addressbook is a precious source of information, they're used to store and carry around documents and often used as access unit to corporate networks. Phones operate in an higly trusted enviroment, pheraps too much trusted: users trust their phones, they trust their operator and operators trust themselves both for convenience and compatibility. Phones communications protocol and networks are closed and etherogeneus and, finally, the hardware landscapes and software platforms of phones are fragmented, preventing a common breeding ground for security development.

These issues have been explored in depth both on the communication side by the works of Paget and Nohl [3] and on the architectural one by those of Pietrosanti [4]. Architectural issues are quite interesting: chatting and texting are predominant operations in the world of a phone, so mobile keyboards adapted their layout to ease the insertion of common spoken words. This will generally degrades the strength of a password generated using such keyboards as numbers and non alphabetical symbols are quite difficult to type

into. Screen dimensions also play an important role in such an enviroment: mobile phones browsers narrow url bar greatly improves phishing attacks, while checking invalid SSL certificates could be either a really difficult task or nearly impossible.

When it comes to mobile phones operating system security we can see too many different implementation strategies, all of them with strengths and weaknesses. Application permissions are generally configurable, unfortunately, the most common solution is an "all or nothing" approach, while a granular permissions fine tuning would be much more safe and interesting.

Finally, on the communication side, there're still too many unsafe protocol in use. As we've just seen, GSM encryption has been cracked but it's not the only protocol who's suffering: SMS is still being used a lot and, yet, heavily vulnerable; from sender spoofing to rogue provisioning the Short Messaging System is not to be considered a secure protocol at all.

When we focus our attention on the privacy side, mobile phones still shows their young age: we've seen how rapid their growth has been, nowadays phones are full fledged computers, carrying lots of personal data. Only some of the available operating systems are capable of offering some form of data encryption [5] [6], for the rest the only choice is to store data in clear.

3 Tor On Mobile Phones And Other Strange Devices

This year, Tor was ported to a great number of mobile phones and some strange device also. Everything started in December 2009 with Tor being run as a bridge on a Chumby One [7], with an official announcement on the Tor Project blog just some months later, around mid February [8].

Even if it's an amusing device, the Chumby One is not a mobile phone at all. The first announcend working port on such devices has been the Nokia N900 [9] [10] which received a Tor port for the Maemo platform in late February.

Next, at the beginning of March, came Tor for Android devices [11] [12]. This one has been a real breakthrough: Android is an operating system for mobile phones with a growing market share, porting the program on such a platform will surely help Tor diffusion and adoptance.

Implementing the Tor program on mobile phones is not an easy task at all: the etherogenity of platforms is the first problem to take into account. If the new hosting platform is following the UNIX standards, then the porting process will be much more easier than rewriting the code from scratch to

adapt the program to the new enviroment. Next problem is the processor power, even if modern phones can sustain an heavy load of work, keeping the CPU up with cryptographic functions is a performance and battery killer. Last, but not least, problem resides in the user interface: as we've seen before in Section 2, the user interface is often narrow and crippled, that's why such port of Tor have to adapt their layout in order to fit in small enviroment and yet be powerful.

4 Tor On The Chumby One

As introduced in Section 3, the Chumby One was the first exotic device to receive a working port of Tor. Chumby multimedia hubs are hackable Linux devices, powered by an ARM cpu and 64MB of RAM: they're an ideal device for running low-powered and low-bandwidth Tor nodes.

The port has been hacked up by bunnie from bunnie:studios and Jacob Appelbaum from TorProject. It was announced on 30th December 2009 from bunnie's blog [7] but it has only been officially accepted into the Tor source tree some months later, the 21st February 2010 [8].

This port is very interesting for many aspects: first, it's quite easy to install. After installing the Chumby ARM cross toolchain, it's just a matter of downloading torproject.org Chumby sources [13] and issuing a "make" command inside the source folder. This will produce a zipped build, unpacking it in the root of an USB key and rebooting the Chumby One with such key inserted will finally install Tor on the device. If a user doesn't want to fiddle with the command line and the cross compilers, unpacking one of the officially provided builds will just be enough to get Tor on the device.

Second, this port is a real working examples on how to port Tor on hardware with limited resources: the Chumby One has a good processor for embedded devices and the minimum required amount of RAM to run a Tor node. Nevertheless, it's currently able to act as a bridge and providing all the multimedia entertainment it was designed to without suffering any issues or slowdown.

There're also some drawbacks but I'm finding them useful to understand how such devices can handle a working port of Tor. First one, the installer will create a swap file for the Chumby One if not already present: this is needed in case the node will start routing a lot of traffic in order to prevent the underlying operating system to crash because of the consumption of all the available RAM.

Second, the Chumby One operating system does not provide an easy to use updating mechanism for unsupported third party software: Tor upgrades

have to rely on the user will to keep the installed program up to date.

Third, the default configuration will set up the node to act as bridge, listening on port 443. This is an important choice since it will both increase the number of nodes for helping people stuck inside Tor-hostile networks and it will prevent the program to eat too many resources too.

Currently the Chumby Tor port is being actively developed and maintained, one of the next interesting features has yet been unveiled by bunnie:studios: it turns out that an easter egg present in the official firmware can activate unofficial support for 3G dongles [14], allowing a Chumby device to route Tor traffic even over the cellular data networks.

5 Tor On Maemo And The Nokia N900

As stated in Section 3, the N900 was the first mobile phone to get a working port of Tor with a graphical controller application.

The Maemo platform is already providing support for Tor users as a third party community site [15], the N900 is the choosen platform for developing a graphical controller application for such operating system.

Installing the Maemo and N900 port is quite easy for this platform too: the user has just to add the already present, but disabled, Extras-devel repository to the software manager, looking for Tor in the newly added packages and reboot the phone. Unfortunately, such repository is marked as "dangerous" even from the Maemo Cummunity site [16], which means the user has the choice to keep this repo enabled, at the risk of having a non functional operating system if an upgrade will go wrong, or enabling it just for installing Tor and subsequent updates which, then, will have to be tracked by hand.

Once installed, the controller application is available from the status menu: selecting the "The Onion Router" icon will bring up the configuration menu where the user can enable or disable the client.

This port is being actively developed but still quite young as the only option, for now, is the choice of wheter or not activating the client functionality for the Tor network.

6 Orbot: Tor On Android

Orbot is the latest official TorProject port of Tor for a mobile platform. This port targets mobile phones shipping with Android firmwares, both for version 1.x and 2.x.

Orbot is not yet available in the Android Market, however its installation

is one of the easier seen so far: just by scanning the QR code from the project page, the user can install the program on his phone [12].

This port is one of the most complete: it ships a copy of Tor, libevent and privoxy, providing HTTP and SOCKS 4a/5 access to the network. The controller application can also set lot of different properties, behaving much like Vidalia [17].

To successfully use the Tor network, Android 1.x users have to download and install from Android Market the ProxySurf web browser and the Beem instant messaging applications, while Android 2.x ones can rely on general system settings. Another option, for both firmwares, is to root the device, in this case Orbot will automatically transparent proxy all TCP traffic.

Development on this port is going strong but what is still missing is a trusted secure browser. However, there's an ongoing effort in porting Mozilla Fennec over to this operating system [18], this will open the road to a port of TorButton [19] for the mobile version of Mozilla browser, which could bring to the community the first secure mobile browser for anonymous communications.

7 MobileTor: Tor On The iPhone and iPod Touch Platforms

iPhone and iPod Touch devices are great mobile platforms: they offer quite good computing power, nice multimedia hardware and a responsive operating system, all packed in a small, portable form factor [20] [21]. It's no surprise that such devices are getting a bigger slice of the growing mobile marketshare. The growth and diffusions of these products is also due to the availability of a continuously growing application marketplace known as *App Store* [22] and an underground, live, development community built around *Cydia* [23] [24].

Choosing between the official route using the Apple iPhone SDK [25] or the underground one with the open source development toolchain [26] [27] is not an easy task: both of them have some pros and cons. When I started looking into them for developing a port of Tor on the iPhone I had to make the choice and went down for the open source road. The outcome was in part forced by the stringent rules for applications submissions to the App Store which prevents submitting new daemons so, for the initial testing and development, it was the open source one.

The first port of Tor on the iPhone platform was done by cjacker huang in December 2007 [28]: he patched the program to have it build and run

under first versions of the iPhone firmware together with a working port of privoxy and he also provided iTor.app, a graphical controller application. Unfortunately, some time later, he disappered together with iTor.app source code and binaries: only his patches, accepted and merged into Tor source tree, survived the event. In February 2010 I began my work from what he left: I polished his own patches as no more necessaries with the growth of both firmware versions and Tor code base and I start offering an up to date, working Tor port for the iPhone again.

My currently working setup includes a Slackware Linux 13.0 64bit open source toolchain built against iPhone OS version 3.1.2 and a local Telesphoreo [29] checkout. Packages are built following Jay Freeman packaging conventions for Cydia and hosted at my own online repository available at `http://sid77.slackware.it/iphone/`.

Right now, the first phase of the project is completed: we finally have a full working port of the command line version of Tor being able to run on iPhone and iPod Touch devices. The program can both be used as an entry point for the Tor network, as a traffic relaying node either over wireless or cellular data networks and as an host for hidden services too.

The second phase of development is going on: even if the port is working well, it can only be used via an SSH connection from a computer or directly on the device using MobileTerminal [30]. These solutions are quite inappropriate for the average user and there's need for a graphical controller application for Tor on the iPhone. A first approach is to implement an SBSettings [31] switch: the user will still have to upload or edit on the device a working configuration but there will be no more need for the command line interface in order to start and stop Tor, just a tap on the appropriate icon. Such a program is ready and soon available in my repository under the name of *Tor Toggle*. A second, more complete, approach is the writing of a Vidalia-like [17] application: this is the best solution for controlling and managing the behavior of Tor on such devices but it's still under heavy development and not yet ready for publication.

Even if Tor on the iPhone is growing well, there're some areas which still need to be addressed. First of all, there's lack of a Tor-secure browser: the iPhone and iPod Touch are currently running Mobile Safari or WebKit based browsers only, tests need to be run to examine such enviroments and possibly ensure a secure anonymous browsing experience as much as it will be allowed by the platform. Another issue is the ability to only set an HTTP proxy from the wireless preference panel: SOCKS proxy are left out. Even if this annoyance is easily bypassed by providing a working polipo port, using Tor as plain SOCKS proxy could have been interesting. Last, the biggest stopper is the inability to set a proxy for the cellular network: the only way

to do so, for now, is to plug the phone in a VPN and then setting a proxy from there, plain VPN-less cellular data connections can not be proxied yet. All of these problems are strictly related to the platform and operating system but they yet impact the adoption of Tor on such mobile devices.

8 Conclusions

On the mobile communications front, Tor has been ported on different, exotic and unusual platforms, such as the Chumby One, the Nokia N900 and Android-based devices.

My work has been focused in getting Tor running on the iPhone and iPod touch and it's currently working very well. What it's still needed is a good, secure, browser for anonymous communications on such platforms, this could be either a result of a good securing work for browsers already available or a newer one written from scratch. Finally, a graphical controller application is yet to be written in order to help with adoption and diffusion of this program.

References

[1] The Tor Project. https://www.torproject.org/.

[2] Running a Tor relay. https://www.torproject.org/docs/tor-doc-relay.html.en.

[3] Chris Paget, Karsten Nohl. GSM: SRSLY? http://events.ccc.de/congress/2009/Fahrplan/events/3654.en.html.

[4] Fabio Pietrosanti. Mobile Security. *Security Summit*, 2010.

[5] BlackBerry Help Center. Encryption. http://docs.blackberry.com/en/smartphone_users/deliverables/1487/Encryption_34117_11.jsp.

[6] BlackBerry Help Center. About content protection. http://docs.blackberry.com/en/smartphone_users/deliverables/1487/About_content_protection_29009_11.jsp.

[7] bunnie:studios. Tor Bridge on chumby One. http://www.bunniestudios.com/blog/?p=800.

[8] Jacob Appelbaum. Chumby One and running a bridge. http://archives.seul.org/or/talk/Feb-2010/msg00261.html.

[9] Jacob Appelbaum. Tor on the Nokia N900 (Maemo) GSM telephone. https://blog.torproject.org/blog/tor-nokia-n900-maemo-gsm-telephone.

[10] The Tor Project. Tor: N900 Instructions. https://www.torproject.org/docs/N900.html.

[11] Jacob Appelbaum. Tor on Android. https://blog.torproject.org/blog/tor-android.

[12] The Tor Project. Tor: Android Instructions. https://www.torproject.org/docs/android.html.

[13] The Tor Project. Chumby Tor sources. https://svn.torproject.org/svn/projects/chumby/.

[14] bunnie:studios. Make Your Own 3G Router. http://www.bunniestudios.com/blog/?p=1076.

[15] Maemo Community. Tor. http://maemo.org/packages/view/tor/.

[16] Maemo Community. Extras-devel. http://wiki.maemo.org/Extras-devel.

[17] The Tor Project. Vidalia. http://www.torproject.org/vidalia/.

[18] Mozilla Wiki. Android. https://wiki.mozilla.org/Android.

[19] TorButton. https://www.torproject.org/torbutton/.

[20] Apple. Apple - iPhone - Technical Specifications. http://www.apple.com/iphone/specs.html.

[21] Apple. Apple - iPod Touch - Technical Specifications for iPod Touch. http://www.apple.com/ipodtouch/specs.html.

[22] Apple. Apple - iPhone - Download thousand of iPhone applications. http://www.apple.com/iphone/apps-for-iphone/.

[23] Jay Freeman (saurik). Cydia. http://cydia.saurik.com/.

[24] Jay Freeman (saurik). Bringing Debian APT to the iPhone. http://www.saurik.com/id/1.

[25] Apple. iPhone SDK. http://developer.apple.com/iphone/.

[26] Jay Freeman (saurik). Upgrading the iPhone Toolchain. http://www.saurik.com/id/4.

[27] iphonedevonlinux. http://code.google.com/p/iphonedevonlinux/.

[28] cjacker huang. Tor and privoxy had been ported to iphone and works very well. http://archives.seul.org/or/dev/Dec-2007/msg00023.html.

[29] Jay Freeman (saurik). Telesphoreo Tangelo. http://www.telesphoreo.org/.

[30] Mobile Terminal. http://code.google.com/p/mobileterminal/.

[31] BigBoss. The Future of BossPrefs. http://thebigboss.org/2008/10/19/the-future-of-bossprefs/.