

Android Security Test

Home to Logging Test App

CarrierIQ Part 2

Carrier IQ Information – Part #2

Written by Trevor Eckhart

[Watch Carrier IQ Video](#)

Video Contents:

Part 1: 0:00 – Device setup

Part 2: 3:15 – Where we don't see CIQ

Part 3: 5:05 – Finding CIQ Application

Part 4

8:34 – Watching Carrier IQ Watch Us

8:39 – Keypresses

12:27 – Receiving a SMS Message

13:35 – Using Browser on WiFi

Part 5: 15:45 – Carrier IQ on an out of service device

Part 6: 16:49 – Conclusions



Carrier IQ believes some of my statements may cause confusion, so I would like to back up my research with more supporting details. All logs posted here were created using standard Android tools. My research focuses on HTC Android devices, but Carrier IQ is on many other devices, too, and may use information differently in those contexts.

I am looking to Carrier IQ for answers here and not HTC because of how many devices this is on. I just happen to use HTC devices, which is why I discovered the application on those phones.

Let's talk about rootkits

The Carrier IQ application as shown in the "stock clients" zip file is how the program looks before it's modified to carrier specifications and is potentially useful to carriers. For end users the stock client indicates that it's running by displaying an icon in the status bar and is contained in a single APK file. We have never seen this version of the CIQ used in the "real world." and to be clear is what fits my definition of a "rootkit"

Here's how Wikipedia defines a "rootkit." I will take the relevant parts step by step from Wikipedia-

A **rootkit** is **software** that enables continued **privileged access** to a computer while actively hiding its presence from administrators by subverting standard **operating system** functionality or other applications.

The way Carrier IQ works, as seen in the training documents ([see last article here](#)), is by enabling someone continued privileged access to our computers (which are Android devices). The application is hidden in nearly every part of our phones, including the kernel (source – <http://htcdev.com/devcenter/downloads>).

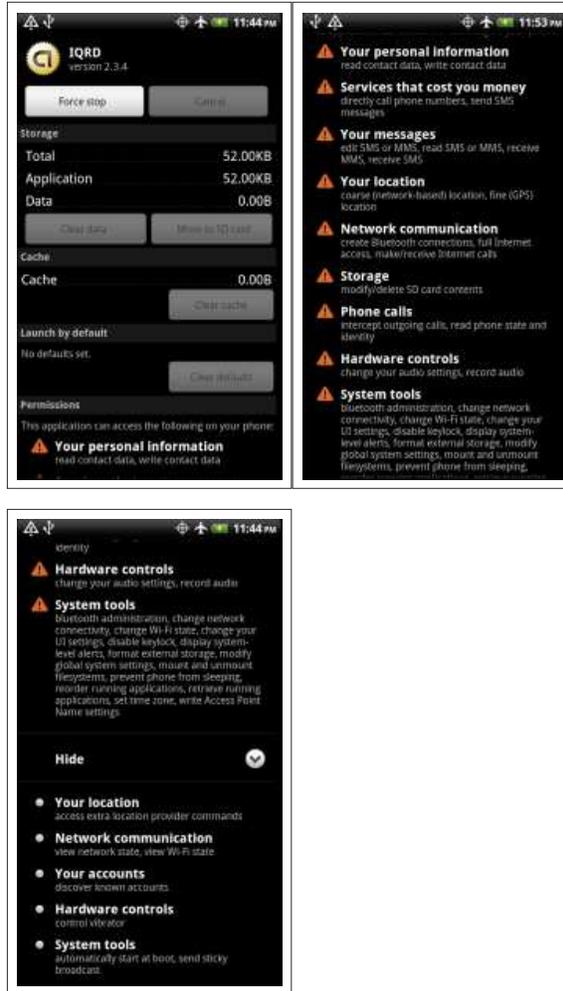
Carrier IQ also subverts standard operating system functionality. For any application, I believe standard operating functionality includes having a descriptively named application; a launcher icon, settings menu, widget, or other method to allow the end user to access the application; and a privacy policy clearly available on the device the application is installed on. Also, as seen in the video, only an application named HTC IQAgent is displayed as a running application on my HTC device. A second program called IQRD never makes itself known as a running application.



It's almost impossible for users to find off switches, user interfaces, policies, or references to IQRD anywhere on the phone. Using standard functionality, the only place you can see that the application is installed on the phone is in Menu -> Settings -> Manage Applications -> All, then scroll down to IQRD. This application has a non-descript icon and offers no information about itself. Even on old devices, IQRD runs continuously because it's set to start automatically at boot. The only option you have to stop the application is to select "force stop"—which does nothing. The application continues to run. This is all particularly concerning when Carrier IQ publicly states that "When Carrier IQ's products are deployed, data gathering is done in a way where the end user is informed or involved."

<http://carrieriq.com/company/privacy.htm>

The very extensive list of Android security permissions granted to IQRD would raise anyone's eyebrow, considering that it's remotely controlled software, but some things such as reading contact data, Services that cost you money, reading/edit/sending sms, recording audio(?!?!?) and writing/changing wireless settings seem a bit excessive (see full list in screenshots below). Even all this is not everything the user has apparently agreed to. IQRD and HTC IQAgent application (shown below) talk to other root-running binaries, and other APK's (such as browser) talk to these two applications and kernel locations. IQRD is able to see actions outside of its own application this way, something I don't see any other apps capable of doing at this level. It's even in the browser looking at data such as HTTPs sessions (see below section for logs).



The second and more obviously named application – HTC IQ Agent – shows no permissions required. This application has an “about” button that shows an HTC logo but no privacy policy or information explaining what the program is. When HTC was asked about it – the company said users need to understand third-party privacy policies.

Third-Party Software: (from HTC <http://www.xda-developers.com/android/htc-responds-once-again/>)

It is also important to note that the phones we build are a compilation of not only software and services from HTC, but also from third parties. These third-party applications and services, such as Carrier IQ (CIQ) and Google Check-in, serve to further improve the customer experience and have their own privacy policies. We encourage consumers to understand the specific policies of any application or service that is enabled on their device.

If HTC's privacy policy doesn't cover the information collected by Carrier IQ, it's unclear whose privacy policy does. Carrier IQ has a minimal privacy policy (<http://carrieriq.com/company/privacy.htm>), but it says, "Our products are designed and configured to work within the privacy policies of our end customers[.]" So whose policy covers this data — Carrier IQ, or the phone manufacturer, or the carrier? Nobody knows for sure.



The only choice we have to “opt out” of this data collection is to root our devices because every part of the multi-headed CIQ application is embedded into low-level, locked regions of the phones. Even if you unlock your device and remove the base application with a sophisticated removal method, neutered, leftover code called from other applications will likely throw an error each time an old action is triggered.

The second part of the definition from wikipedia:

The term *rootkit* is a concatenation of “root” (the traditional name of the privileged account on Unix operating systems) and the word “kit” (which refers to the software components that implement the tool).

CarrierIQ runs the binaries as user root in our ramdisk. The Carrier IQ code is in almost every application: browser, dialer, SMS, media player, the kernel itself, who knows where else. Please read more about the kernel below.

```

init.shooter.rc
301
302 # HTC IPC server
303 service htcipcd /system/bin/htcipcd
304     user root
305     group root system
306
307 # CIQ backend daemon
308 service iqcd /system/bin/iqcd
309     user root
310     group root system
311
312 # CIQ frontend daemon
313 service iqfd /system/bin/iqfd
314     user root
315     group root system
    
```

From wikipedia -

Rootkit detection is difficult because a rootkit may be able to subvert the software that is intended to find it. Detection methods include using an alternative, trusted operating system; behavioral-based methods; signature scanning; difference scanning; and memory dump analysis. Removal can be complicated or practically impossible, especially in cases where the rootkit resides in the kernel; reinstallation of the operating system may be the only available solution to the problem.

In real-world usage, it's very hard for an average user to even be aware of Carrier IQ. To see it, we need to look in low-level file systems, kernel source code, and system logcat logs. It has no launcher icon, no settings, no program to opt in to. Even DRM, which is typically another "hidden service", has a notice about activation when you use HTC Watch.



you are able to decline if you do not want to use the DRM service, but you can not decline to use Carrier IQ.

It's almost impossible to fully remove Carrier IQ. The browser is modified to send to Carrier IQ daemon, as is almost everything else. The application is so deeply embedded in our devices that a user must rebuild the whole device (system.img and boot.img) directly from source code to remove every part of CIQ.

Devices out of Contract especially have an issue

IQ Insight Experience Manager uses data directly from the mobile device to give a precise view of how the services and the applications are being used, even if the phone is not communicating with the network. (From http://www.carrieriq.com/company/PR.Experience_Manager.CTIA-09.090325.pdf)

*Such **profile transmission to the SQC 402 residing on the target device(s) may be achieved using any of a variety of transport mechanisms and standards including Short Message Service ("SMS"), Hypertext Transport Protocol ("HTTP"), Hypertext Transport Protocol Secure ("HTTPS"), Wireless Application Protocol ("WAP") Push, IP-based Over-the-Air (IOTA) protocol, OMA/DM, or other protocols that are known in the art or that may be developed in the future. (From <http://www.patents.com/us-7609650.html>)***

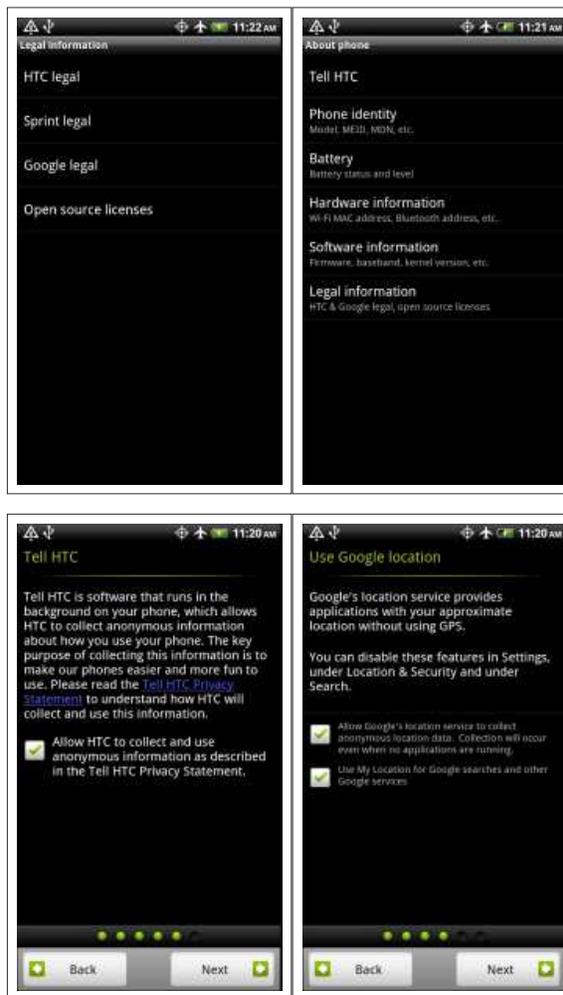
But Carrier IQ's Woods said that her company's software is set to disable data collection if the device's SIM card or mobile carrier changes. (From – <http://www.informationweek.com/news/security/mobile/231903096?pgno=2>)

There are a few problems with all of this. There are no SIM cards on CDMA phones – CDMA users have no options to take a device completely off a network. It's not common for CDMA devices to change carriers (without cloning a device, which is probably frowned upon in most cases). Every time I get a new phone, I stick the old one in airplane mode with wifi, then activate the new phone. On the old phones CIQ collectors are still shown to be running in the background passing data around, and there's no way to stop and remove them (the same logcat logs as above are visible). Furthermore, they're looking to the URL in iqprofile.pro, which is an HTTPS address.

Developers are constantly getting new devices to make apps, themes, etc. Regular users buy new devices like the HTC EVO 3d

just to play with innovative new features. Android devices are linux computers with hardware that (sometimes) includes a phone radio. In short, there are tons of ways to use these devices other than as a phone. Sometimes users don't sign a service contract with anyone, never turn on the cell radio, and use the device exclusively on wifi.

Below are the only opt-in switches/legal terms on my HTC device. In the statement in the above section, HTC says we must understand that third-party software is not HTC software, and is subject to third-party terms. Where are Carrier IQ's terms and policies for allowing this application to run and collect data — even if it's just local — on any device out of a carrier's service? As we saw from the permissions requested by the IQRD application, the program is able to log very sensitive user data, including reading contacts/SMS, recording audio, modifying network connections and more.



What do we see Carrier IQ actually doing?

Before I begin, remember this analysis is specific to Android and HTC. There are other devices and platforms that could use the Carrier IQ API differently; this is just what I know Carrier IQ is capable of doing on an HTC device.

Let's start with what's seen when the IQD binary application runs. The only text we see is about using PCAP (<https://en.wikipedia.org/wiki/Pcap>)

```
# iqd
IQMetricsPCAPThread_Start – setting up PCAP link
pcap_thread
pcap_open_live -> 494640,113
IQ_InitializeBoosterPackIP -> 0
```

On HTC devices, the easiest way to watch the rest of Carrier IQ is to run a logcat. We can see two identifiers (AgentService_J and HTC_SUBMITTER_C) doing most of the work, but this is only a brief look at what's happening, and doesn't reflect everything that might be going on or all data being looked at

SECURITY ALERT The interesting thing is because we are able to see this happening in logcat, anything with the right permissions can see the same thing. It means programs other than CIQ, such as crash reporting software or any app that can read logs, will also be able to see the same exact logs.

Webpage visited –

```
V/AgentService_J( 713):
Action[1021]:com.htc.android.iqagent.action.al34
I/HTC_SUBMITTER_C( 713): (0)
submitAL34:-1155628198,https://www.google.com/
V/AgentService_J( 713):
(0)dwPageID:1321694298970,szURL:https://www.google.com/
```

Location Statistics – *(seems to trigger whenever location updates or get queried)*

Intent – com.htc.android.iqagent.action.lc30

```
V/AgentService_J( 716):
Action[157]:com.htc.android.iqagent.action.lc30
I/HTC_SUBMITTER_C( 716): (516010663)
submitLC30:234,40752055,-73806468,911,113,0,516010663
V/AgentService_J( 716):
(0)lc30_TStamp_lo:516010663,lc30_Tstamp_hi:234,lc30_Latitude:40752055,lc
```

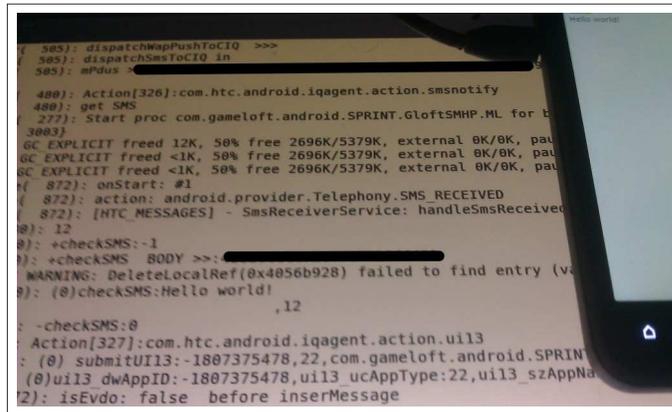
Media Statistics – (from Test UI)

Intent – com.htc.android.iqagent.action.mp03

```
V/AgentService_J( 723):
Action[787]:com.htc.android.iqagent.action.mp03
V/AgentService_J( 723): ErrorCode:103
V/AgentService_J( 723): NumTracks:3
V/AgentService_J( 723): mp03_dwSize:201
V/AgentService_J( 723): mp03_dwPktRcvd:202
V/AgentService_J( 723): mp03_dwPktDup:203
V/AgentService_J( 723): mp03_dwPktLoss:204
V/AgentService_J( 723): mp03_dwPktSent:205
V/AgentService_J( 723): mp03_dwAvgJitter:206
V/AgentService_J( 723): mp03_dwAvgLatency:207
V/AgentService_J( 723): mp03_wAvgRate:208
V/AgentService_J( 723): mp03_ucCodec:209
V/AgentService_J( 723): mp03_ucPad:210
I/HTC_SUBMITTER_C( 723): (0)
submitMP03:103,3,sizeof(arrStats):96
```

SMS Received –

Intent – com.htc.android.iqagent.action.smsnotify

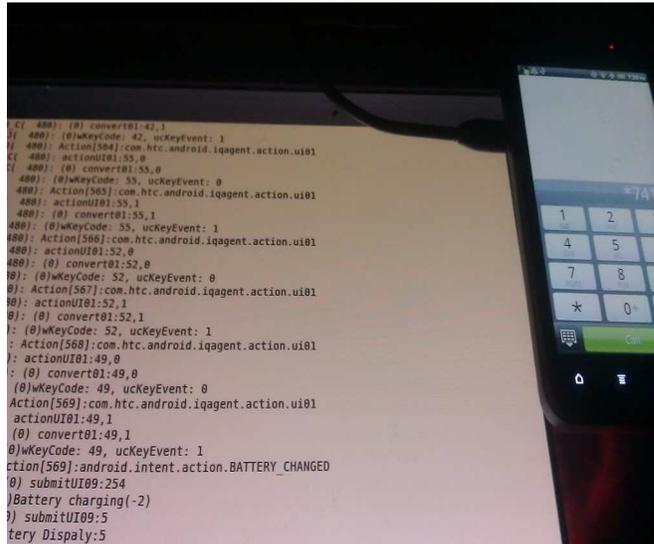


```
D/SMSDispatcher( 2464): dispatchWapPushToCIQ >>>
D/SMSDispatcher( 2464): dispatchWapPushToCIQ >>>
D/SMSDispatcher( 2464): dispatchSmsToCIQ in
D/SMSDispatcher( 2464): mPdu >[a message pdu is in hex(?)
here, removed]
V/AgentService_J( 713):
Action[877]:com.htc.android.iqagent.action.smsnotify
V/AgentService_J( 713): get SMS
V/AgentService_J( 713): 43
V/AgentService_J( 713): +checkSMS:-1
```

```
V/AgentService_J( 713): +checkSMS BODY >>: [a message body  
is here(hex??), removed]  
I/HTC_SUBMITTER_C( 713): (0)checkSMS:testing123 [the  
contents of the message sent]  
I/HTC_SUBMITTER_C( 713): hii  
I/HTC_SUBMITTER_C( 713): (this is my message)EWT,48
```

Keypress made -

Intent – com.htc.android.iqagent.action.ui01 –



**Pressed 1: (wkeycode 49), you can see ucKeyEvent = 0 when
key is pressed**

```
V/AgentService_J( 716):  
Action[170]:com.htc.android.iqagent.action.ui01  
I/HTC_SUBMITTER_C( 716): actionUI01:49,0  
I/HTC_SUBMITTER_C( 716): (0) convert01:49,0  
V/AgentService_J( 716): (0)wKeyCode: 49, ucKeyEvent: 0
```

ucKeyEvent = 1 when released

```
V/AgentService_J( 716):  
Action[171]:com.htc.android.iqagent.action.ui01  
I/HTC_SUBMITTER_C( 716): actionUI01:49,1  
I/HTC_SUBMITTER_C( 716): (0) convert01:49,1  
V/AgentService_J( 716): (0)wKeyCode: 49, ucKeyEvent: 1
```

I pressed the button 2: (wkeycode 50)

V/AgentService_J(716):
Action[182]:com.htc.android.iqagent.action.ui01
I/HTC_SUBMITTER_C(716): actionUI01:50,0
I/HTC_SUBMITTER_C(716): (0) convert01:50,0
V/AgentService_J(716): (0)wKeyCode: 50, ucKeyEvent: 0

V/AgentService_J(716):
Action[183]:com.htc.android.iqagent.action.ui01
I/HTC_SUBMITTER_C(716): actionUI01:50,1
I/HTC_SUBMITTER_C(716): (0) convert01:50,1
V/AgentService_J(716): (0)wKeyCode: 50, ucKeyEvent: 1

I press the button 3: (wkeycode 51) (cut off key released from here on, you get the point by now)

V/AgentService_J(716):
Action[191]:com.htc.android.iqagent.action.ui01
I/HTC_SUBMITTER_C(716): actionUI01:51,0
I/HTC_SUBMITTER_C(716): (0) convert01:51,0
V/AgentService_J(716): (0)wKeyCode: 51, ucKeyEvent: 0

Button 4 (wkeycode 52):

V/AgentService_J(716):
Action[196]:com.htc.android.iqagent.action.ui01
I/HTC_SUBMITTER_C(716): actionUI01:52,0
I/HTC_SUBMITTER_C(716): (0) convert01:52,0
V/AgentService_J(716): (0)wKeyCode: 52, ucKeyEvent: 0

Home button pressed (wkeycode 11)

V/AgentService_J(713):
Action[528]:com.htc.android.iqagent.action.ui01
I/HTC_SUBMITTER_C(713): actionUI01:11,0
I/HTC_SUBMITTER_C(713): (0) convert01:11,0
V/AgentService_J(713): (0)wKeyCode: 11, ucKeyEvent: 0

Back Button pressed: (wkeycode 27)

V/AgentService_J(713):
Action[554]:com.htc.android.iqagent.action.ui01
I/HTC_SUBMITTER_C(713): actionUI01:27,0
I/HTC_SUBMITTER_C(713): (0) convert01:27,0
V/AgentService_J(713): (0)wKeyCode: 27, ucKeyEvent: 0

Screen On/Off –

Intent – com.htc.android.iqagent.action.ui02

Screen On –

V/AgentService_J(717):
Action[355]:com.htc.android.iqagent.action.ui02
I/HTC_SUBMITTER_C(717): (0) submitUI02:1,0,0
V/AgentService_J(717):
(0)OldUIState:1,NewUIState:0,UIEvent:0

Screen Off –

V/AgentService_J(717):
Action[357]:com.htc.android.iqagent.action.ui02
I/HTC_SUBMITTER_C(717): (0) submitUI02:0,1,0
V/AgentService_J(717):
(0)OldUIState:0,NewUIState:1,UIEvent:0

Signal Changes –

V/AgentService_J(713):
Action[935]:com.htc.android.iqagent.action.ui08
I/HTC_SUBMITTER_C(713): actionUI08 metric:6, 3
V/AgentService_J(713): (0)ASU, TECH:6, 3

Battery Usage Changes – (yes the typo is in logcat not me) 😊

I/HTC_SUBMITTER_C(713): (0) submitUI09:5
V/AgentService_J(713): Battery Dispaly:5

Application Opened Intent – com.htc.android.iqagent.action.ui15

Application Focused:

Intent – com.htc.android.iqagent.action.ui19–

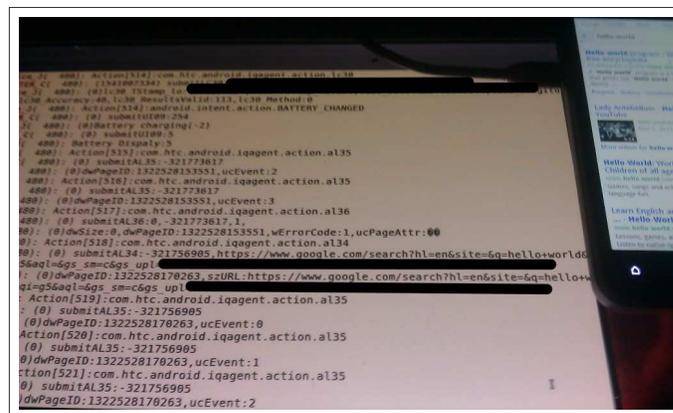
V/AgentService_J(713):
Action[561]:com.htc.android.iqagent.action.ui19
I/HTC_SUBMITTER_C(713): (0) submitUI19:-1725705692,0
V/AgentService_J(713):
(0)ui19_dwAppID:-1725705692,ui19_ucFocusEvent:0
V/AgentService_J(713):
Action[562]:com.htc.android.iqagent.action.ui19
I/HTC_SUBMITTER_C(713): (0) submitUI19:-1725705692,0
V/AgentService_J(713):
(0)ui19_dwAppID:-1725705692,ui19_ucFocusEvent:0

Now let's talk about HTTPS

In the online world, HTTPS is pretty much the only thing protecting sensitive data moving around. In a nutshell, when you first connect to <https://www.yourbank.com>, the browser checks SSL certificates and makes sure the site operator is who it says it is. After that, all traffic between the user and the site is encrypted.

From <https://blog.httpwatch.com/2009/02/20/how-secure-are-query-strings-over-https/>, we see an example of this. HTTPS strings + data after the SSL handshake really can't be sniffed outside of browser.

Now, although it's insecure, HTTPS usernames/passwords CAN be passed in plain text. So while my examples are based on a google search, if you did <https://mysite.com?username=MYNAME&password=MYPASS>, that exact string would be passed into the CIQ application.



Googling Hello World over SSL:

```
V/AgentService_J( 713):
Action[1035]:com.htc.android.iqagent.action.al34
I/HTC_SUBMITTER_C( 713): (0)
submitAL34:-1155376757,https://www.google.com/search?hl=en&
site=&q=hello+world&oq=hello+world&aq=f&aqi=g5&aql=&gs_sm=e&
gs_upl=[removed some sensitive looking information]
V/AgentService_J( 713):
(0)dwPageID:1321694550411,szURL:https://www.google.com
/search?hl=en&site=&q=hello+world&oq=hello+world&aq=f&aqi=g5&
aql=&gs_sm=e&gs_upl=[removed some sensitive looking
information]
```

Below are a few excerpts from logcat that show what Carrier IQ was doing while I logged into PayPal. I did not feel comfortable posting more information than this, since the login strings Carrier IQ grabbed were detailed. Notice it even says **(from:com.android.browser)** showing that Carrier IQ has been integrated directly into the browser's code. The application is reading not only HTTP, but the HTTP's details about the page I visited down to the JS(**javascript**) and CSS(**Cascading Style Sheets**) files which are all the "background code" that control how webpages look and feel.

```
V/AgentService_J( 468):
Action[1328]:com.htc.android.iqagent.action.nt10
I/HTC_SUBMITTER_C( 468): (-3)
actionNT10:0,-1,304,4,0,0,4,https://www.paypalobjects.com
/en_US/i/pui/core/nav_sprite.gif
V/AgentService_J( 468):
(-3)Size:0,SocketID:-1,Type:304,AppType:0Mode:4,URI:https://www.paypalobje
/en_US/i/pui/core/nav_sprite.gif(from:com.android.browser)
V/AgentService_J( 468):
Action[1328]:com.htc.android.iqagent.action.nt10
I/HTC_SUBMITTER_C( 468): (-3)
actionNT10:0,-1,304,4,0,0,4,https://www.paypalobjects.com
/en_US/i/icon/icon_load_roundcorner_lock1_186x42_withlock.gif
V/AgentService_J( 468):
(-3)Size:0,SocketID:-1,Type:304,AppType:0Mode:4,URI:https://www.paypalobje
/en_US/i/icon
/icon_load_roundcorner_lock1_186x42_withlock.gif(from:com.android.browser)
V/AgentService_J( 468):
Action[1329]:com.htc.android.iqagent.action.nt10
I/HTC_SUBMITTER_C( 468): (-3)
actionNT10:0,-1,304,4,0,0,4,https://www.paypal.com/en_US
/i/logo/paypal_logo.gif
V/AgentService_J( 468):
(-3)Size:0,SocketID:-1,Type:304,AppType:0Mode:4,URI:https://www.paypal.cor
/en_US/i/logo/paypal_logo.gif(from:com.android.browser)

V/AgentService_J( 468):
Action[1315]:com.htc.android.iqagent.action.nt0f
I/HTC_SUBMITTER_C( 468): (-3)
actionNT0F:722,-1,0,5,0,0,4,https://www.paypalobjects.com
/WEBSCR-[SOME_UNIQUEID]/css/core/global.css
V/AgentService_J( 468):
(-3)Size:722,SocketID:-1,Code:0,AppType:0Mode:4,URI:https://www.paypalobje
/WEBSCR-[SOME_UNIQUEID]/css/core
/global.css(from:com.android.browser)

V/AgentService_J( 468):
Action[1328]:com.htc.android.iqagent.action.nt10
```

```
I/HTC_SUBMITTER_C( 468): (-3)
actionNT10:0,-1,304,4,0,0,4,https://www.paypalobjects.com
/WEBSCR-[SOME_UNIQUEID]/js/lib/min/widgets.js
V/AgentService_J( 468):
(-3)Size:0,SocketID:-1,Type:304,AppType:0Mode:4,URI:https://www.paypalobje
/WEBSCR-[SOME_UNIQUEID]/js/lib
/min/widgets.js(from:com.android.browser)
```

My Conclusions

I have shown what the Carrier IQ application is capable of doing on an HTC device. The fact that it's embedded into the shipped device raises very serious security and privacy concerns. My original article was intended to be my take on the entire process with enough information for anyone to verify. I cited my sources throughout the article and mirrored training files to support my findings. Everything I wrote continues to be to the best of my knowledge. **Many people are clearly confused about this application and what it does, and it's being explained to nobody.**

1. The CIQ application is embedded so deeply in the device that it can't be fully removed without rebuilding the phone from source code. This is only possible for a user with advanced skills and a FULLY unlocked device. Even where a device is out of contract, there is no OFF switch to stop the application from gathering data. The files and code littered across many protected operating system locations will ALWAYS be there, and we can see logcat proof of it running — wasting CPU cycles, PMEM space, and whatever else. Regardless, the applications should not be allowed to run and collect data outside of *possibly* a contract with a carrier. Any user who wants a full removal method should have one.
2. The CIQ application is receiving not only HTTP strings directly from browser, but also HTTPS strings. HTTPS data is the only thing protecting much of the “secure” Internet. Queries of what you search, HTTPS plain text login strings (yuck, but yes), even exact details of objects on page are shown in the JS/CSS/GIF files above — and can be seen going into the CIQ application.
3. The CIQ portal is not anonymous if devices upload packages by equipment id and other identifying metrics and can individually be tasked packages, as we saw in the training document. ([see previous article](#)) I would like to know exactly who has seen this data, what data has been recorded, and who has recorded it. This data should also be subject to some clear privacy policy. The only existing privacy policy we can find rings hollow when we know the software logs sensitive identifiable data:
 - A. *Our data gathering and data storage policies are built from industry best practice. Our products allow us to*

address privacy & security requirements that vary country-by-country and customer-by-customer. There are a variety of techniques involved in protection of privacy and in implementation of security policy, including anonymization of certain user-identifiable data, aggregation of data and encryption of data, etc. (From <http://carrieriq.com/company/privacy.htm>)

4. If a bad actor discovered a vulnerability or used malware, he could potentially exploit that opportunity to become a “CIQ operator,” leaving many users helpless against the extensive collection and misuse of their own information and no way to stop it. With so much moving code across the operating system, I would say the chances of malware looking here isn’t that far-fetched.

An application should never be this hard to fully remove for security reasons—especially out of contract—when it serves no good purpose for the user, and its use should be opt-in ONLY.
