Hacking TomTom GO

Christian Daniel, Thomas Kleffel

6.12.2005

Zusammenfassung

Als im Sommer 2004 das Navigationssystem *TomTom GO* am Markt erschien, kam sehr schnell der Verdacht auf, daß das Gerät auf dem Betriebssystem Linux basiert. Im Laufe der kommenden Wochen traten wir diesen Beweis an und die Arbeiten dazu werden in diesem Vortrag vorgestellt.

Neben der Dokumentation der GPL-Verletzung, die TomTom letztlich zur Offenlegung der Kernel-Sourcen zwang, wurde auch der Signaturmechanismus für die Bootfiles nachgebaut und damit die Plattform vollkommen geöffnet. Die Möglichkeit, nun einen eigenen Kernel booten und eigene Anwendungen ausführen zu können, führte zur Gründung des OpenTom-Projektes. Dort wurde der alte Kernel durch die aktuelle Version 2.6 ersetzt und ein eigener, freier SD-Karten-Treiber entwickelt.

TomTom reagierte auf die Abmahnung durch Harald Welte von GPL-Violations.org wirklich vorbildlich: Nicht nur alle GPL-relevanten Sourcen wurden offengelegt, sondern TomTom sah auch sofort den Wert der Community ein – und arbeitet mit ihr nun konstant zusammen.

1 Motivation

Natürlich kann man mit diesen Navigationssystemen ganz prima von A über B nach C fahren – aber für einen Hacker ist der offensichtliche Nutzen eines Gerätes natürlich immer nur halb so faszinierend wie die weiteren Möglichkeiten. Mit Linux als Betriebssystem lädt die Hardware geradezu zum Spielen ein: MP3-Player, Video-Spieler, Rückfahr-Kamera... – alles das ist möglich sobald eigene Anwendungen ausgeführt werden können. Mit Farbbildschirm, Touchscreen, Bluetooth, GPS, USB, einem großen, leistungsfähigen Lautsprecher und Akkukapazität für knapp vier Stunden Betrieb ist eigentlich alles dran, was man sich von so einem schicken "Embedded System" wünschen kann.

Die Treiber für alle[1] TomTom-spezifischen Hardwarekomponenten sind zusammen mit den verwendeten Kernelquellen unter [1] zu bekommen. Auch die Firma Naviflash behauptet, ein "stabiles LINUX-Betriebssystem" einzusetzen. Leider sind die von Naviflash veröffentlichten Quellen unvollständig und die Hardware lange nicht so interessant, wie dies bei TomTom der Fall ist.

2 Der Hack

2.1 Reverse Engineering

Im August 2004 hielten wir das erste *TomTom GO* in Händen, und hatten nur eine vage Vermutung, daß die Software im Inneren auf Linux basierte. Um diese Vermutung zu beweisen, gingen wir auf die Suche nach dem System: Zuerst wurde die beigelegte SD-Karte genauer untersucht, wobei sich schnell herausstellte, daß das System offensichtlich von dort geladen wird. Nach einiger Zeit gelang es tatsächlich, aus einer Datei mit dem Namen 'system' einen gepackten Linux-2.4-Kernel und die dazugehörige Init-Ramdisk zu extrahieren. Damit war der Weg frei, TomTom zur Herausgabe der Kernel-Quellen zu zwingen. Unser Ziel war jedoch, einen eigenen Kernel – wenn möglich sogar Version 2.6 – zu booten.

[1] Fast alle... der SD-Karten-Treiber liegt nur als Objektcode vor, allerdings wurde ein freier Treiber entwickelt, der unter [4] verfügbar ist.

Um vernünftig entwickeln zu können war daher das Auffinden einer seriellen Konsole notwendig. Nach einigem Herumprobieren fanden wir diese dann auch auf zwei Pins der Anschlußleiste auf der Unterseite des Geräts – einfacher als erwartet. Der erforderliche Pegel-Wandler für das mit 3.3V arbeitende Gerät war schnell gebaut und wir konnten dem Kernel nun beim Booten zuschauen.

Leider wartete dort, obwohl das Image auf Busybox basiert, keine Shell auf uns. Die weitere Analyse der Init-Ramdisk ergab, daß sich die Shell nur bei gesetztem Debug-Flag aktiviert ist und TomTom dieses Flag in einer Release-Version nicht setzt. Wir änderten die Init-Ramdisk also entsprechend ab und packten sie wieder in die ttsystem-Datei zusammen. Leider weigerte sich der Bootloader, das von uns modifizierte System zu starten.

Nach kurzer Ratlosigkeit – der Bootloader befindet sich im Flash auf der Platine – öffneten wir unser TomTom. Allerdings war die Idee, das Flash auszulesen und so an den Bootloader zu kommen, nach einem kurzen Blick auf die eng bestückte Platine gestorben. Hier kam uns ein von TomTom veröffentlichtes Software-Update gerade recht, da es eine Datei enthielt, die sich als Update für den Bootloader entpuppte – die genaue Untersuchung konnte also losgehen. Da Disassemblierung juristisch ein heißes Pflaster ist, mußten wir uns für die Veröffentlichung von OpenTom einen anderen Weg überlegen[2]: Wir durchsuchten den Bootloader nach Konstantentabellen für gängige Hashing-Verfahren und wurden bei MD5 fündig. Bereits früher waren uns 16 scheinbar nutzlose Bytes am Ende jedes Abschnitts der 'ttsystem'-Datei aufgefallen – deren Zweck war damit klar.

Anhand des Orginal-Image versuchten wir nun herauszufinden, über welche Teile von 'ttsystem' sich der Hash erstreckt, jedoch ergab keiner unserer Versuche einen korrekten Wert. Der MD5-Hash wird also noch weiter verändert, bevor er im Image abgelegt wird. Wieder griffen wir mit bekannten Konstantentabellen an und wurden wieder fündig – diesmal mit der Blowfish-Referenzimplementation. Es lag auf der Hand, daß der zugehörige Schlüssel irgendwo in den 256kB des Bootloaders stehen muß. Es ergaben sich also (262.144 - 16) Möglichkeiten, die natürlich recht schnell durchprobiert waren. Ergebnis: Der MD5-Hash war tatsächlich nochmals per Blowfish verschlüsselt und der Key war jetzt bekannt[3].

Mit diesem Wissen konnten wir nun unser modifiziertes System booten und zum ersten Mal eine Shell auf dem GO öffnen. Kurz darauf konnten wir das System auch nach unseren Wünschen zusammenstellen und hatten z.B. mit madplay einen tragbaren MP3-Player[4].

Einige Zeit später gelang es Christian auch, Linux 2.6 auf die Plattform zu portieren – die Entwicklung eigener Treiber für die Peripherie war dann aber nicht mehr nötig, da TomTom inzwischen die Quellen offengelegt hatte.

2.2 Überzeugungsarbeit bei TomTom

Leider zeigte TomTom auf unsere Bitte um die Kernel-Quellen unter Hinweis auf die GPL, wie viele andere Firmen auch, keine Reaktion. Daher baten wir Harald Welte, den Gründer des GPL-Violations-Projekts [2], die Sache weiter zu verfolgen. Er konnte TomTom mit einer einstweiligen Verfügung auf die Problematik aufmerksam machen[5], indem einem großen Versandhändler verboten wurde, das Gerät in Deutschland zu verkaufen.

Die Reaktion der Firma TomTom war schließlich mehr als vorbildlich: Die Quellen und alle Informationen, die zum Bauen eigener Images notwendig sind, wurden kurzfristig offengelegt, Harald Welte und Christian Daniel nach Holland eingeladen und dem CCC ein großzügiger Betrag gespendet[3]. Die ganze Sache war in wenigen Tagen erledigt. Seitdem pflegt TomTom eine gute Beziehung mit der Community, von der alle Beteiligten profitieren können. Thomas Kleffel und Christian Daniel durften an der Entwicklung der Nachfolgemodelle mitarbeiten und bekamen so eine Anschubfinanzierung für ihre eigene Embedded-Linux-Firma.

[2] Was nicht heißt, daß wir nicht trotzdem auf diesem Weg auf ein paar Ideen gekommen sind...

[3] Nachdem die Aktion uns einge schlaflose Nächte gekostet hat, soll unsere Beute hier nun auch genannnt werden: 0xD8,0x88,0xd3,0x13,0xed,0x83,0xba,0xad,0x9c,0xf4,0x1b,0x50,0xb3,0x43,0xfa,0xdd

[4] Es ist wirklich erstaunlich, wie gut der eingebaute Lautsprecher funktioniert. Besonders die Bässe sind – da das Gehäuse als Resonanzkörper funktioniert – recht überzeugend.

[5] Da es schwierig ist, aus Deutschland eine Abmahnung in den Niederlanden zu erwirken, mußte TomTom indirekt zur Herausgabe der Quellen gezwungen werden.

	GO	GO 300	GO 500	GO 700	ONE	Rider
Codename	Classic	M100	M300	M500	Bilbao	Glasgow
Erschienen	Q3 2004	Q2 2005	Q3 2005	Q3 2005	Q4 2005	Q4 2005
CPU	S3C2410	S3C2410	S3C2440	S3C2440	S3C2440	S3C2440
Speed	200MHz	200MHz	380MHz	380MHz	380MHz	380MHz
RAM	32 MB	32 MB	32 MB	64 MB	32 MB	32 MB
Storage	SD/MMC	SD/MMC	SD/MMC	HDD	SD/MMC	SD/MMC
GPS	SIRF 2	SIRF 2	SIRF 2	SIRF 2	SIRF 3	SIRF 3
Sound	LS/Dock	LS/Dock	LS/Dock	LS/Dock	LS/Buchse	Buchse
Bluetooth	nein	ja	ja	ja	ja	ja
USB Host	ja	ja	ja	nein	ja	ja
Mic-In	nein	nein	ja	ja	nein	ja
Remote	nein	nein	ja	ja	nein	nein
ttyS0	ja	ja	ja	ja	nein	nein

Tabelle 1: Hardwareübersicht

3 Hardware

Die erste Hardware, auf der auch der beschriebene Hack basierte, erschien im Sommer 2004. Im Frühjahr/Sommer 2005 folgten dann *GO 300*, *GO 500* und *GO 700* und aktuell sind die Modelle *ONE* und *Rider* erschienen. Die genauen technischen Daten sind in Tabelle 1 zusammengefasst.

Für unsere Zwecke eignet sich das *GO 500* am besten. Es besitzt die schnellere CPU, Bluetooth, Sound mit Lautsprecher und Mikrofoneingang, der USB-Port ist auf Host-Betrieb umschaltbar und das ttyS0 ist über den Docking-Port zugänglich. Leider gehört es mit einem Straßenpreis von knapp 500 Euro zum eher teureren Spielzeug...

Deutlich günstiger sind *GO Classic* und *GO 300* zu haben; sie sind bis auf das beim *GO Classic* fehlende Bluetooth identisch und ähnlich gut geeignet. Die CPU ist zwar deutlich langsamer, dafür ist ein gebrauchtes *GO Classic* schon ab 200 Euro zu haben.

Das Spitzenmodell *GO 700* ist zum Basteln nur eingeschränkt zu empfehlen, da der USB-Port hier nur zu einer USB-IDE-Bridge, nicht aber zur CPU führt. Wer sich allerdings feine SMD-Lötarbeit zutraut und keinen Wert auf die Garantie legt, kann das Gerät öffnen und den USB-Host nachrüsten.

Bei den Modellen *ONE* und *Rider* ist der serielle Port von außen nicht zugänglich. Das ist nicht zu schlimm, schließlich kann nach dem Booten des aktuellen OpenTom-Images auch per Bluetooth auf das Gerät zugegriffen werden. Entwicklung am Kernel dürfte sich so allerdings etwas schwierig gestalten.

3.1 USB-Host

Der externe USB-Port aller Modelle (bis auf den des *GO 700*) wird ursprünglich nur vom Bootloader als USB-Device genutzt. Ist das Gerät mit einem USB-Host verbunden, wird die eingesteckte SD-Karte via USB-Storage freigegeben. Dieser Mechanismus dient dazu, Karten- und Softwareupdates bequem vom PC aus einzuspielen.

Viel interessanter ist die Tatsache, daß sich die entsprechenden Pins an der CPU softwareseitig auf den auf dem Chip vorhandenen USB-Root-Hub verbinden lassen (siehe [10, 11]) und damit externe USB-Geräte vom System aus angesprochen werden können. Laut USB-Spezifikation [12] werden jedoch USB-Host und -Device extern unterschiedlich beschaltet. Im USB-Host werden beide Signalleitungen (DP und DM) mit 15kΩ-Widerständen gegen Masse gezogen. Das USB-Device zieht nur DP mit 1.5kΩ gegen Vcc (3.3V).

Da die Hardware nur für die Verwendung als USB Device vorgesehen ist, ist dort auch nur der 1.5kΩ-Widerstand gegen 3.3V vorhanden. Dieser ist glücklicherweise abschaltbar[6].

[6] Bei einigen älteren *GO Classic*-Modellen fehlt der zuständige Transistor. Dort muß im anzuschließenden USB-Device (am besten ein alter Hub, an den dann unmodifizierte Geräte angeschlossen werden können) der 1.5kΩ-Pullup-Widerstand entfernt werden.

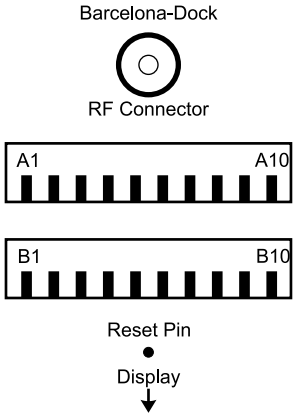

Abbildung 1: Barcelona Dock

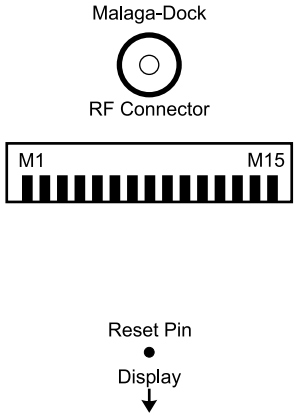

Abbildung 2: Malaga Dock

Pin	Signal	Pin	Signal	Pin	Signal
A1	ttyS0 TxD (3.3V)	B1	Signal ground	M1	Signal ground
A2	ttyS0 RxD (3.3V)	B2	Signal ground	M2	Dock sense
A3	ttyS0 RtS (3.3V)	B3	Dock sense	M3	Dock sense
A4	ttyS0 CtS (3.3V)	B4	Dock sense	M4	Mic-In mono
A5	Signal ground	B5	Line-Out left	M5	Line-Out left
A6	JTAG RST	B6	Line-Out right	M6	Line-Out right
A7	JTAG TMS	B7	Car-radio mute	M7	Car-radio mute
A8	JTAG TCK	B8	Ignition sense	M8	Light sense
A9	JTAG TDI	B9	+5V from dock	M9	Ignition sense
A10	JTAG TDO	B10	+5V from dock	M10	ttyS0 TxD (3.3V)
				M11	ttyS0 RxD (3.3V)
				M12	ttyS0 RtS (3.3V)
				M13	ttyS0 CtS (3.3V)
				M14	+3.3V from device
				M15	+5V from dock

Tabelle 2: Barcelona und Malaga Dock Pinout

Um den Stecker als USB-Host-Port zu nutzen ist ein spezielles Kabel notwendig, das am einen Ende einen Mini-B-Stecker (Buchse am TomTom), am anderen Ende einen B-Stecker (Buchse am USB-Gerät) und außerdem die beiden 15kΩ-Widerstände enthält. Mit etwas Lötkolbengeschick ist das aber recht schnell von Hand angefertigt.

3.2 Docking-Port

Das *GO Classic* hat im Gehäuseboden zwei Buchsen mit je zehn Pins (Barcelona-Dock, Abbildung 1). Die Modelle *GO 300*, *GO 500* und *GO 700* besitzen nur eine Buchse mit 15 Pins (Malaga-Port, Abbildung 2). Beide Typen enthalten einen Audio-Ausgang (in Stereo!), einen seriellen Port, einen 5V-Eingang für die externe Stromversorgung und IO-Pins für die Zündung und Autoradio-Mute. Am Barcelona-Dock liegt zusätzlich noch der JTAG-Port der CPU an. *GO 500* und *GO 700* haben einen Mikrofoneingang für die Freisprecheinrichtung.

Leider sind die verwendeten Stecker des Herstellers Yamahichi nicht einzeln zu bekommen. Wer nicht seinen mitgelieferten Dock zerlegen oder einen zusätzlichen kaufen möchte, kann sich passende Stecker mit etwas Aufwand selbst herstellen. Eine Anleitung dazu findet sich unter [7].

3.3 Der serielle Port

Bei allen GOs (nicht *Rider* und *ONE*) ist am Docking-Port eine serielle Schnittstelle herausgeführt (siehe auch Abbildungen 1 und 2). Vorhanden sind die Signale RXD, TXD, CTS und RTS, verwendet werden allerdings nur TXD und RXD[7]. Der Port findet sich im System als /dev/ttyS0 wieder und dient dem Kernel als Boot-Konsole.

Da die Signale direkt zur CPU führen, haben sie einen Pegel von nur 3.3V. Damit das Gerät beim Verbinden mit einem PC nicht zerstört wird, muß dieser Pegel auf RS232-Niveau umgewandelt werden, was z.b. ein MAX3232 erledigt. Im Internet und unter [5] sind dazu verschiedene Schaltungen zu finden. Eine andere Möglichkeit ist das Ausschlachten eines Handy-Kabels. Diese sind für alte Modelle billig zu bekommen und enthalten meist den benötigten Pegelwandler.

4 Bootkonzept

Im 256kB großen Flash-Speicher des Gerätes sind nur ein proprietärer Bootloader und einige Konfig-Informationen[8] enthalten. Dieser Bootloader erlaubt den Zugriff auf das jeweilige Speichermedium per USB-Storage. Wenn kein USB-Host angeschlossen ist, sucht der Bootloader eine Datei mit dem Namen 'system', lädt und startet diese.

Auf den meisten Geräten handelt es sich dabei um ein kleines Tool, das den Bootloader bei Bedarf auf den neuesten Stand bringt[9] und danach das eigentliche System aus der Datei 'ttsystem' startet. In dieser Datei befindet sich sowohl der Kernel, wie auch die Init-Ramdisk.

Die Dateien folgen dem gleichen Aufbau aus einzelnen Abschnitten, an deren Ende sich die Signatur (MD5-Summe des Abschnitts, mit Blowfish und bekanntem TomTom-Key signiert) anschließt. Der genaue Aufbau dieser Datei ist in [6, 8] dokumentiert. Unter [4] sind Tools zum Erzeugen und Zerlegen dieser Dateien erhältlich.

5 Kernel

Der von TomTom veröffentlichte Kernel basiert momentan auf Version 2.6.13. Die meisten Treiber (Touchscreen, Sound, GPIO, ...) befinden sich unter im Verzeichnis drivers/barcelona[10]. Dort befindet sich auch eine Hardwareabstraktion, die die unterschiedlichen Modelle auf einen Nenner bringt.

Suspend-to-RAM ist gut unterstützt, da es selber von der Navigationssoftware benötigt wird. Die meisten Treiber exportieren einfache Interfaces in /dev und sind über ioctls zu steuern. Der Bildschirm wird als Framebuffer angesprochen; es ist sogar möglich mit einem MPlayer Videos auf dem Gerät abzuspielen, allerdings stößt hier die CPU schnell an ihre Grenzen.

Es ist genug RAM vorhanden, um kleine eigene Anwendungen auch parallel zur Navigationssoftware laufen zu lassen.

6 OpenTom

Viele Details über Hard- und Software der TomTom-Geräte sind im Rahmen des OpenTom-Projekts unter [4] dokumentiert. Wir laden alle Interessierten ein, ihre Erkentnisse und Ideen im Wiki niederzuschreiben. Auch ein Teil des TomTom-SDK, das es erlaubt, z.B. die Menüstruktur der Navigationsanwendung zu verändern und eigene Programme einzubinden, ist dort dokumentiert.

Weiterhin ist dort auch ein Image erhältlich, mit dem es möglich ist, per Bluetooth eine Netzwerkverbindung zum TomTom aufzubauen. Per Telnet und FTP können dann eigene Programme auf

[7] Warum auf dem Port auch CTS und RTS verfügbar sind, ist unklar. Die Signalleitungen werden vom Linux-Treiber für den entsprchenden UART nicht unterstützt.

[8] Seriennummer, Bluetooth-Adresse, Touch-Screen-Kalibration, etc...

[9] Wenn ein Softwareupdate einen neuen Bootloader enthält, wird einfach eine neue 'system'-Datei auf die SD-Karte kopiert. Beim nächsten Start wird der Bootloader dann automatisch auf den neuesten Stand gebracht.

[10] 'Barcelona' ist der interne Projektname für das erste Modell.

das Gerät kopiert und ausgeführt werden ohne, daß an der Hardware gebastelt werden muss (und die Garantie bleibt dabei auch erhalten).

Nicht zuletzt gibt es ein MPlayer-Image, mit dem entsprechend heruntergerechnete DivX- und MPEG-Videos abgespielt werden können.

7 Über die Autoren

Christian Daniel (27) und Thomas Kleffel (23) sind beide bereits seit jungen Jahren fasziniert von technischen Geräten in jeglicher Form. Seit einigen Jahren treffen sie sich hin und wieder, um an neuen Geräten einen Mehrwert zu entdecken oder auch selber etwas auf die Beine zu stellen. Im Frühjahr 2005 entschlossen sich die beiden Informatik-Studenten, zusammen mit Matthias Kleffel eine eigene Firma zu gründen. Als *maintech GmbH* [9] sind sie seitdem für einige große Firmen in Deutschland und Europa tätig. Neben der Arbeit als Embedded Linux Consultants entwickeln sie auch eigene Produkte in den Bereichen Netzwerktechnik und Industrieautomatisierung.

8 Danksagung

Wir möchten uns bei allen Freunden, die uns bei diesem Projekt mit Rat und Tat zur Seite standen, herzlich bedanken. Insbesondere Harald Welte für die Durchführung der rechtlichen Dinge und Matthias Kleffel für das Entdecken des Blowfish-Algorithmus.

Thomas besonderer Dank geht an Teresa, die die Kraft für dieses Projekt und noch viel mehr gab! Danke für die schöne Zeit mir dir!

Christian dankt seinem Schatz Sabine – wie wären die durchgehackten Nächte nur ohne so viel Verständnis möglich :-)

Und natürlich geht unser Dank und Gruß an Peter-Frans, Ayal, Serhiy, Dimitry, Jeroen, Johan und den Rest des Entwicklungsteams bei TomTom.

Literatur

[1] `http://www.tomtom.com/gpl`

[2] `http://www.gpl-violations.org`

[3] `http://www.gpl-violations.org/news/20041024-linux-tomtom.html`

[4] `http://www.opentom.org`

[5] `http://www.opentom.org/index.php/Serial_Console`

[6] `http://www.opentom.org/index.php/Ttsystem_images`

[7] `http://www.opentom.org/index.php/Homemade_plugs`

[8] `http://www.franken.de/de/veranstaltungen/kongress/2004/04-3-2-tomtomgo.pdf`

[9] `http://www.maintech.de/`

[10] `http://www.embedon.com/pdff/S3C2410A%20datasheet.rar`

[11] `http://www.embedon.com/pdff/S3C2440A%20datasheet.rar`

[12] `http://www.usb.org/developers/docs/usb_20_02212005.zip`