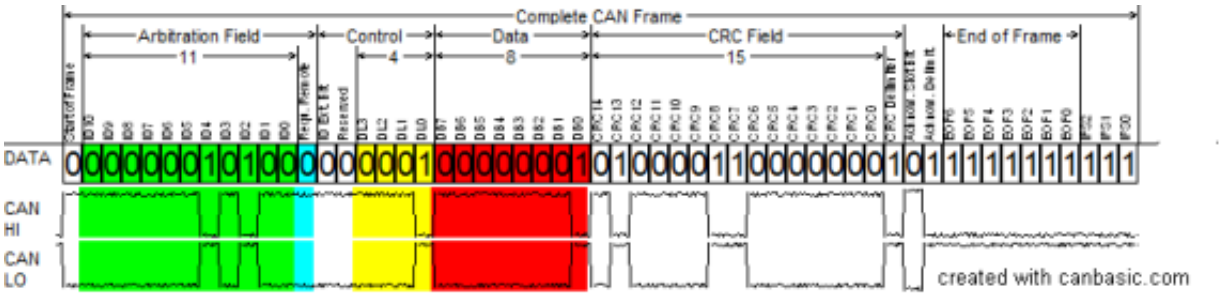
# CAN Hacking: Introductions

October 21, 2013 by Eric Evenchick 47 Comments




We're introducing a new series on CAN and automotive hacking. First, we'll introduce CAN and discuss how in-vehicle networks work.

In 1986, Bosch introduced the Controller Area Network protocol. It was designed specifically for in-vehicle networks between automotive controllers. CAN became a popular option for networking controllers in automotive, industrial, and robotics applications. Starting in 2008, all vehicles sold in the US must use CAN.

Modern vehicles are distributed control systems, with controllers designed to handle specific tasks. For example, a door control module would take care of locks and windows. CAN allows these controllers to communicate. It also allows for external systems to perform diagnostic tasks by connecting to the in-vehicle network.

Some examples of CAN communication in a vehicle include:

- The engine control module sending the current engine speed to the instrument cluster, where it is displayed on a tachometer.
- The driver's door controller sending a message to another door controller to actuate

the window.
- A firmware upgrade for a controller, sent from a diagnostics tool.

CAN is usually used with little or no security, except for the obscurity of the communications. We can use CAN to USB interfaces to listen to the traffic, and then decode it. We can also use these tools to send forged messages, or to perform diagnostic actions. Unfortunately, most of the tools for dealing with CAN are proprietary, and very expensive. The diagnostics protocols are standards, but not open ones. They must be purchased from the International Organization for Standardization.

Next time, we'll get into the structure of CAN frames, and how traffic is encoded on the bus.

[Image via Wikipedia]

# CAN Hacking

- Introductions
- The In-vehicle Network
- CAN Protocols (planned for 10/29/13)
- Building CAN Hardware (planned for 11/5/13)

DIY soda can battery  What are the best hacking documentaries?
Inside The Clapper  Ask Hackaday: Hacking lingo fails
How to setup a Hackerspace from someone who's done it before

---

Filed Under: Featured, Network Hacks  Tagged With: automotive, CAN, CAN Hacking

# Comments

sqelch says:
October 21, 2013 at 5:05 pm

Awesome

Reply Report comment

Nikola says:
October 21, 2013 at 5:06 pm

FINALLY! I'm going to school for embedded systems and CAN has been something that has peaked my interest lately. I haven't gotten good info from professors or the web. This is a topic of interest for me. Glad to see this going on!

Reply Report comment

Wolfy says:
October 22, 2013 at 3:36 am

It's spelled piqued.

Reply Report comment

Reset says:
October 22, 2013 at 4:39 am

Not, if its a play words. The Peak-Adapter is a prevail USB device to de and encode can messages.

Reply Report comment

Reset says:
October 22, 2013 at 4:42 am

By the way, CAN was not invented by BOSCH. BOSCH had bought it from a small company at Wolfendbuettel in Germany. I do not remember the name.

Reply Report comment

Nick @ Hack Directory says:

October 21, 2013 at 5:45 pm

CAN is good for linking ECU's, though has security issues.

Reply *Report comment*

Biomed says:
October 21, 2013 at 5:52 pm

CAN is used all over in X-Ray systems. I'm all ears!

Reply *Report comment*

Hack says:
October 21, 2013 at 5:54 pm

CAN is an amazing protocol. The addressing has built-in prioritization and collision avoidance. The CAN controllers can handle serious failures at the physical layer and still work. It can be used in a way very similar to a client /server architecture, but I think of it as really designed for broadcast traffic. For example Imagine you have a speed sensor that transmits your speed a few times a second. Your speedometer sees that packet and updates the display, the cruise control sees it and adjusts your engine to maintain speed. Your car door listens for a command to lock or unlock. Anything can generate that command... The door switch on any door, the key fob receiver, the alarm module, etc. And those things could listen for those commands also so, for example, the alarm can set itself.

Sorry for the rambling, I've been a fan of can bus in hobby electronics for years and have never understood why more people don't use it.

Tl&dr... Canbus is great!

Reply *Report comment*

Hack says:
October 21, 2013 at 5:57 pm

Add you can get canbus controllers pretty cheap (or free... Microchip samples. And canbus chips can be programmed as simple io expanders that are almost. Set and forget. And how many hobby grade electronics have had the level of development as automotive electronics? Talk about reliable...

Reply *Report comment*

Nimbusgb says:

October 24, 2013 at 9:35 am

Simple really. The bus is stamped all over with proprietary data. You have to pay vast sums of money to get stuff certified. Hobbyists are often frustrated inventors and would like to exploit a good idea if they come up with one. The barriers to complete entry are simply too high.

Reply *Report comment*

**Andrew Becker says:**
October 21, 2013 at 6:26 pm

CAN is also used in aerospace applications.

Reply *Report comment*

**strider_mt2k says:**
October 21, 2013 at 6:42 pm

Fantastic!
I have a 2012 Jeep Wrangler that I would like to get to know at this level.
THANK YOU FOR POSTING THIS!!

I purchased a Bluetooth to ODBII adapter and own a copy of Torque for Android, but I've never gone beyond reading some codes and resetting my wife's car once when it threw a code.
I eagerly await more installments and welcome and other references anyone could throw out there for the curious noob such as myself!

Reply *Report comment*

**Michael says:**
October 21, 2013 at 6:54 pm

I'm also excited to see this. Among other things, I've been wanting to develop my own keyless entry for my 2005 CRV (if anyone has one with the keyless entry module, please sniff the protocol for me!)

This is going to be fun!

Reply *Report comment*

**damennix says:**

October 21, 2013 at 6:57 pm

Ok you have my attention. I like how you added the dates for future posts…. But I want them NOW!!! such a tease

Reply Report comment

**Cabe (@drgncabe) says:**
October 21, 2013 at 8:58 pm

Nice, I look forward to the next posts. I've been playing around on the PCI bus in my '99 Jeep for a while (to the point where I can issue some commands) but just now getting into K+CAN with my car once my cable arrives. Some of the programs are in German though (German engineering/German tech programs). I can do a lot more (programming wise) with the car, it's almost like it's made to hack!

Reply Report comment

**Mike says:**
October 21, 2013 at 10:08 pm

I'd really like to hack into my 2005 BMW K1200LT. Does anybody know if it has the CAN bus? I have the repair manual, but can't tell based on the schematics.
thanks.
Mike

Reply Report comment

**Galane says:**
October 21, 2013 at 10:27 pm

As a person who has worked on cars since the 1970's – I don't much like this CAN and other overly complicated and overly expensive systems.

There's no need to have a computer inserted into the dead simple electrical circuit of a switch and the lights or a switch and a window motor.

I remember from in the 80's when multiplexed wiring etc "in the future" would make the electrical system in a vehicle simpler (HA!) and less expensive (double HA!).

What we got instead was "body control modules" pointlessly electronically controlling all the stuff that worked perfectly without any silicon – other than the glass in the windows and light bulbs.

Now with the 2010+ Camaro it's gotten so stupid that this is built into the *radio*.

Remove the radio and *nothing works*. Can't even so much as unlock the doors, turn on the lights, open a window or even start the engine. If you want an aftermarket radio the only solution is an expensive replacement control box that duplicates all the non-audio/radio functions of the system built into the stock radio.

If a vehicle manufacturer wants to build a least cost car, the easiest way would be to limit the computer control to only the drivetrain and let the rest get by on SIMPLE and CHEAP wires and switches like it did for 100+ years before this computers running everything nonsense.

That would be especially useful now that systems have been developed that could be clandestinely wired into a car and have been shown to be able to (with the added equipment) remotely control anything connected to the car's computer.

To safeguard against such attacks will either require additional complexity for self testing to check for unknown or unauthorized equipment connected up somewhere, or drastically simplifying the systems so that any compromising alterations are easy to discover. To make it harder to hack, design program chips with a proprietary pinout and encrypted data with a complex, multistep access/handshake protocol to get the program and the hardware to talk to each other.

The car hacking demonstrations have demonstrated how lax security is on these systems, and the horrendous stupidity of the systems being made so pointlessly complex.

My vote is for simplifying!

Reply *Report comment*

**Trui says:**
October 21, 2013 at 10:38 pm

And really, what was wrong with horses in the first place ?

Reply *Report comment*

**Phrewfuf says:**
October 21, 2013 at 10:52 pm

Top of my hat to you, this comment pretty much sums it up.

Reply *Report comment*

**0xfred says:**
October 22, 2013 at 5:05 am

+1. As I was reading the above ramble, that's almost word-for-word the comment that was forming in my head.

Reply *Report comment*

**Mr .Ed says:**
October 23, 2013 at 9:00 am

It's fun to read the first car ads printed in newspapers, back when car dealers were still competing with horses for buyer's attention.

Some of the ads mentioned how clean and odorless they were (I mean the cars!) and how you didn't have to feed them (also the cars). But they left out the fact that if you owned a couple of cars they would never suddenly produce a brand new car that would also be yours.

They also apparently didn't anticipate or acknowledge smog and soot and exhaust (or they thought that such odors smelled good vs. road apples and that smog and soot wasn't "dirty," I guess). To this day cars are still among the most significant sources of air pollution in populated areas. And I don't know about those older cars but mine needs to be "fed" regularly from a hose and periodically from little bottles of viscous toxic goo.

Horses also get you home even if you're passed out cold or fall asleep… at night… in a blizzard… and without needing an actual road. The solid and liquid exhausts from horses are also not merely biodegradable, they're compostable into organic fertilizer that helps produce more renewable horse-fuel unlike car exhaust or it's old, toxic fluids. Horses are also immune to solar flares, unlike ECM modules and the grid that powers gas stations and traffic control devices.

Automotive infrastructure also makes having a car virtually mandatory because of how it forces municipalities to physically expand enough (i.e. "sprawl") to accommodate the roads, parking lots, gas stations, dealerships, mechanic shops, tire shops, parts stores and any other auto-related built environment or infrastructure. Horses require a fraction of that support infrastructure. So the catch-22 is that to build a car-friendly city involves making it so big that you will HAVE TO use some kind of vehicle to get around in it.

While I'm not calling for ditching cars and going back to horses, I do think that not all so-called "advances" in technology are necessarily for the best, at least over the long run (factoring in issues and costs related to sustainability tends to change the bottom line -sometimes drastically- which is why most economists don't). Technological advancement will always be a trade-off, so we'd best understand exactly what it is we're getting and what we're giving up to get it. But we do so love our shiny things and blinking lights and stuff.

"Car culture" will almost certainly be considered by historians in the distant future to be a relatively brief flare-up of collective insanity. I have a car myself but that's because we've built ourselves a society where most of us can't afford not to own one. I used to ride a bicycle everywhere but got sick of being hit by cars (twice was plenty) then I moved to a city that's even less bike-friendly than where I got hit both times. Too bad my car never gets together with my neighbor's car so they make us both some new cars; that would be handy.

Still, this CAN stuff is cool and I'm interested in learning more, especially with regards to using it for non-automotive (hobby project) applications.

Reply *Report comment*

**Michael says:**
**October 21, 2013 at 11:44 pm**

I totally agree the cheap cars must be simple, just without any electronics. Back in thos 1970's, there was only wiring, switches, lamps – nothing was easier than changing a light bulb (compared to today's compact cars). – BUT – none of those cars does have a speedometer, computer that shows gas milage etc. Personally I prefer drive one of these modern cars, but do remembrr the (good??) old times when cars where just "mechanical" :-)

Reply *Report comment*

**Mike Szczys says:**
**October 22, 2013 at 8:25 am**

I think the same could be said for my front-load washer. It used to be all electro-mechanical, but my fancy machine has a "hand wash" setting the old ones simply didn't implement and that's a really nice feature.

Reply *Report comment*

**gio says:**
**October 22, 2013 at 12:26 am**

That's one of the reasons why I still drive Fiat 125p straight from the 80s.

Reply *Report comment*

**Remy Dyer** says:
October 22, 2013 at 1:58 am

There is a right way and a wrong way to design embedded systems.

The wrong way centralizes functionality that has no business being together, and makes it opaque to repairmen. This is usually chosen so as to "optimize for cost" by minimizing the number of computers... you know, in a "there's maybe a market for 5 computers globally" kind of way.

The right way is to keep things as simple as will work, preferably not even using OS's, when the tasks can be separated so one function per microcontroller. This way there is no multiplication of complication, and each separate processor's code becomes simple and easily maintained. (and preferably also, kept with the install so that future repairmen have a chance to fix it!). This is known as robust design, engineering for maximum reliability, as a priority far above "cost".

As manufacturers get better at this, expect, nay, demand that they become compliant! An essential, and yet still lacking part of this is the availability of the source, as well as the barrier-to-modification. The auto industry is notorious for keeping all their tech secret – so much so that most "scientifically published, peer reviewed" papers read more like sales brochures with even less actual detail than the patents that they're trying to push.

Unfortunately, it seems keeping source code secret has a reputation as being some kind of "good" engineering practice, even though it's basically fundamentally at odds with science and engineering. If you contract an engineer to make something for you, you will get the blueprints – if they don't provide them, then they didn't engineer it.

Car manufacturers have a lot to learn in this regard, but even more so do the "engineering software" companies who "sell products" in the MSFT style. It's disgusting, and one day people will find it difficult to believe that it was ever considered reasonable business practice.

All that said: CAN is very good – no where near the performance of, say, FireWire, but still exactly as "KISS" like as you could hope for in a low speed / high reliability / easily debugged multidrop communications protocol. (Firewire is perfection at high bandwidth, but carries with it quite an overhead in terms of protocol, it was clearly designed by a comittee of computer engineers obessed with getting every little detail right. It is anarchistic democracy to USB's benevolent dictatorship. CAN is just a few mechanics talking shop, and getting shit done whilst occasionally yelling over the other, but that's ok, because everyone knows the pecking order, and shuts up and listens when it's urgent.)

Apart from sheer BW, CAN shits all over ethernet's stupid "if a collision, wait a random time, then try again" approach to sharing a wire. (Hint: this is why ethernet is not used to share wires anymore – do you know the difference between a "hub" and a "switch"? Remember 10Base2 ?)

Apart from the horribleness putting a "standard" behind a paywall (just like selling software licences imho, anti education.), CAN is a good bit of work, whose actual impact on the world for the better will be proportional to those benefiting from it, and therefore dependant upon being widely understood and used where appropriate. (Again, something that paywalls / licensing / general IP shit destroys).

I'm looking forward to this series :)

Reply *Report comment*

**Trui says:**
October 22, 2013 at 2:32 am

It is a common fallacy that splitting a complex design into a simple components will improve reliability. The reality is that the components may be simple, but the interaction between them becomes complicated.

Reply *Report comment*

**FELLER says:**
October 22, 2013 at 3:19 am

@Trui, I had a mechanical engineer try to apply this methodology to a project. The results were exactly as you described. It was ultimately my conclusion that every division in to smaller segments comes at the cost of interface overhead.

The art is in identifying where borders can be created without paying major overhead costs. You can carve up territories without a disproportionate level of effort dealing with tower of babel problems.

Essentially using good judgement in determining how much to break up the systems.

Unmanned systems frequently weigh these trade-offs when presented with the choice of all-in-one do everything components which are robust & inexpensive by the nature of sharing PCB layers vs separating functionality which promises better upgradability & customization.

Smaller chunks are more flexible, but the temptation to go this route is too frequently based on the fallacy that the system will be expanded on rather

than becoming technologically obsolete as a whole before there is any desire to make incremental improvements.

Increasingly I'm finding my block diagrams are following the trends of the embedded world's SoC's and bringing more and more functionality under the same roof.

@Galane

"If a vehicle manufacturer wants to build a least cost car, the easiest way would be to limit the computer control to only the drivetrain and let the rest get by on SIMPLE and CHEAP wires and switches like it did for 100+ years before this computers running everything nonsense."

It's been my experience that there is nothing cheap about wires and switches unless you are dealing with order quantities in the hundreds of thousands. The documentation overhead is brutal when dealing with elaborate wiring harnesses. Fault troubleshooting is similarly difficult when trying to tease the correct wire number out of a bundle or identify a wire with chaffed insulation.

With CAN-bus you run 2-5 wires along the body of your vehicle and you're done with everything except the last mile of analog, which is frequently much easier to deal with because of it's limited scope.

If your system complexity is enough to involve more than 1 or 2 people in the electrical harnessing you can expect weeks of labor hours to be spent generating schematics that could otherwise be done in a couple of days for a highly digital system. You need a significant BoM differential & order volume to recovery that difference in labor cost. Good digital systems can also self-report faults minimizing the requisite skill level of service technicians and increasing the rate of diagnosing and solving problems.

Reply *Report comment*

**Philip McKenna says:**
**October 22, 2013 at 11:32 am**

Agreed, wiring is not cheap. I typically hear that eliminating a single wire can save $0.30 which is significant in the automotive industry. This is why many switches and other accessories are placed on a bus like LIN. For example steering wheel switches may have 10 or so buttons that would all be hard wired. If you take a simple IO extender IC to read in switch inputs and place results on the bus now you only need to run 3 wires from the module to the controller and the IO extender IC may cost less than $1.

In addition to cost savings, weight is also a factor. All those copper wires add up in the end and with higher and higher demands to increase fuel economy, reduction in weight is also important. There is even consideration to move to aluminum wiring in some cases to reduce overall weight.

**fonz says:**
October 22, 2013 at 12:04 pm

bingo, car manufacturers optimize price to the last cent so you can be sure when
they use CAN/LIN etc. instead of a wire because that is the cheapest and most reliable

**fartface says:**
October 22, 2013 at 7:25 am

" If you want an aftermarket radio the only solution is an expensive replacement control box that duplicates all the non-audio/radio functions of the system built into the stock radio." No, you can bury the stock radio in the dash. Used to do it with the craptastic Buick Rivieras back in the 90's.

Reply *Report comment*

**C Hart says:**
October 22, 2013 at 8:01 am

While I disagree with the majority of this, and would prefer more of my vehicle be "x-by-wire" with a good, standard bus architecture, one point you bring up about the CAN physical layer really irks me as well.
> Remove the radio and *nothing works*.
Having a network become nonoperational when a node is physically disconnected is a major flaw!! CAN has a specific requirement for resistance on the bus, typically relatively high resistance between HIGH and LOW signals at stub nodes, and relatively low resistance at bus termination points. This is something I'd like to learn more about.

Reply *Report comment*

**KWest says:**
October 22, 2013 at 12:20 pm

> Remove the radio and *nothing works*.
I was deep into the car electronics world when this first started with Chevy Cavaliers and then migrated to other models. The CAN network was still operational when the radio was removed, however because BCM didn't see the messages on the bus from the radio there were system errors, nothing serious just enough to throw a CEL. Initially there was the radio in the trunk solution, then there was the chime module that replaced the radios messages on the bus and played the door chime, now we have modules that let you do almost anything a knob, button, switch in the vehicle will do with a simple 2 wire hookup.

Reply *Report comment*

**Richard Harvie (@Richard_Harvie) says:**
**October 22, 2013 at 1:03 am**

As someone who works with CAN for a living, I can tell you that there are certainly security measures on the network. If you want to change anything on your cars ECU then it's unlikely that you'll be able to do it without passing a seed / key test first – and your only chance of figuring that out is to reverse engineer an OEM tool.

If you just want road speed / RPM / engine load etc then you can get that quite easily on most vehicles. Fuel economy is usually absent, so needs calculating from other data. ISO15765 is the spec to look out for – it's a fairly simple request response protocol.

Reply *Report comment*

**Nik Pfirsig says:**
**October 22, 2013 at 9:43 am**

It seems to me that part of the security in automotive can applications is specifically intended to create a customer dependency on the dealer service departments.
In the case of the radio, for example, even if you replaced the GM supplied radio with the same model radio from a junkyard, you have to pay whatever the dealer service dept charges to get the security code reset before it will work. In addition,, some models include alarm system functions in the radio, so without the original radio, nothing works.
When I drop a few kilobuck on something, I like to think of it as mine.

Reply *Report comment*

fonz says:
October 22, 2013 at 12:13 pm

but it also means that the is no reason for any to break in and steal the radio it won't work unless you go to a dealer an get it unlocked for your car

it can also part of the "immobilizer" in that the dash, the key, the ecu etc. has to agree
or the car won't start

Reply Report comment

charliex says:
October 24, 2013 at 11:13 am

As someone who hacks ECUs to bypass these security measures for tuning/aftermarket, most are pretty well known, the harder ones are leaked by staff or extracted from leaked OEM software. And there is a lot of software you can just buy off the shelf. Even with the locked tricore CPU's there is a wealth of info.

J2534 , J1699 are useful docs too.

Most OEMS have their own set of PID's but a lot of them have been figured out.

Also an easy entry to CAN is AT90CAN Atmel + transceiver, Atmel provide a CAN Lib though if you're using 1Mbps CAN it needs to be optimised a bit to keep up. Or a Drewtech Mongoose is a good choice for car CAN.

Reply Report comment

charliex says:
October 24, 2013 at 11:43 am

also as a cheap pre made (about $30)
https://www.olimex.com/Products/AVR/Development/AVR-CAN/

software
http://www.atmel.com/devices/AT90CAN32AUTOMOTIVE.aspx?tab=tools

there are even cheaper ones, but i've used this one.

Personally i'd avoid the Peak CAN, and dearborn ones too. In fact i'd avoid any that are expensive unless you need a true analyser/sniffer, the only reason i go with the mongoose is that i like the drewtech guys, they had a lot to do with the

protocols and a lot of OEM and aftermarket software works with it.

pretty much every car sold in the usa has a j2534 compatible reflash tool , by federal law they're required to be 'encrypted' but its either just the seed/key or some very basic encryption. J2534 itself is widely documented, but when you get that you still need to know how to do the flash, its the difference between sending a UDP packet and a web server.

The OEM's have to provide you as the end user with a way to reflash, they can charge a reasonable fee and a lot will lock it down to a particular reflash tool or set, like the moongose, however its an ASCII check and usually via a registry key, so easy to change. most are 404 now.

Often they'll be a front end for a web back end that queries the VIN/partnumber and then goes and fetches the binary from a sever, others just have it hard coded in to it.

this is all for reflashing the ecu, when it comes to reading dials and stuff, you'll get some support of the basic OBD II , but not all PID's have to be supported, and OEM's are allowed their own types, again these have been commonly documented on the web.

you'll usually get rpm, speed, long term and short term fuel trims etc., you can query the OBD II to get a list of what PID's are supported, look up the j1699 test tool on sourceforge from drewtech too.

still at least its much much better than K Line or ALDL which was so slow as to be unusable, some reports could be seconds out of date.

Reply *Report comment*

**Geebles says:**
**October 22, 2013 at 2:22 am**

I look forward to this :)

Reply *Report comment*

**Keith says:**
**October 22, 2013 at 2:36 am**

I'm into electric vehicles and diy digital dashboards, I've done some research on the topic and there are some good input in these comments. My hopes are to get some

MCUs with CAN and Ethernet for my diy EV. Basically set up the can bus to work as normal while having sensor variables like speed, charge, motor temp, odometer, etc – communicate to the dashboard over a simple ethernet line. I agree with whoever said dedicate a single controller to each task. You want as little processing on these boards as possible, and you want room in memory. Where HPC goes bigger and better to do more, think of CAN as low performance computing to do less. You don't want your controllers doing anything but their role, the simpler it gets, the safer.

This information below enriched my mind about CAN hacking, I hope others enjoy it! illmatics.com/car_hacking.pdf

Reply *Report comment*

**Dave says:**
**October 22, 2013 at 6:38 am**

Been working on a CAN System for about a year. A lot of detail. Software is very picky.

Reply *Report comment*

**fartface says:**
**October 22, 2013 at 7:30 am**

Sparkfun and others have a great arduino canbus shield and libraries already built.

Waiting for the Arduino hate to come my way... How dare I suggest a duino...

Reply *Report comment*

**Sparhawk817 says:**
**October 22, 2013 at 9:32 am**

hatehatehatehatehatehatehate.... anyways, that's probably at least a LITTLE bit part of the intent behind the article. hacking is made easier with 'duino's, why not make ourselves autonomous cars with them?!?(exaggeration i know... but fun thought!)

Reply *Report comment*

**six677 says:**
**October 22, 2013 at 2:48 pm**

can still use the arduino shield with another micro I guess.

Reply *Report comment*

**bortels (@bortels)** says:
**October 22, 2013 at 7:46 am**

I wonder if this could be used to bypass some of the ugly aspects of keyless entry; my Prius keyfob is in it's last days, they're quite expensive to replace/rekey via the dealer (hundreds of dollars), and besides – that's no fun. Hacking the protocol is contraindicated, as it's made to thwart car thieves (and likely beyond my *current* skills) – but if I could plug a USB/CAN adapter in, I might be able to have an obsolete android phone (enough CPU, easy to interface, and sitting idle in drawer being the key factors here) work as a bluetooth/wifi backdoor for keyless entry. Out-of-Band control for the win!

It's always bugged me that adding computers to cars made them *less* hackable, not more – that's not how it should be, and maybe this series can help. Sounds awesome!

Reply *Report comment*

**Philip McKenna** says:
**October 22, 2013 at 11:18 am**

I would love to see a follow up to this introducing the Local Interconnect Network (LIN) Bus which is becoming more and more prevalent in vehicles and is much simpler than CAN. LIN is used for many applications, switches, motor drives, etc... Also I think it would be good to touch on CAN-FD (flexible datarate) which will start showing up in some vehicles as early as 2016-7 and Single Wire CAN (used mostly at GM). CAN-FD supports higher datarates or larger packages of data in a single message, all while being backwards compatible with current CAN. I have also heard some rumbling of OEMs considering to start encrypting CAN messages for to try and circumvent people from exploiting the CAN bus. Looking forward to this series.

Reply *Report comment*

**Alan Kilian** says:
**October 22, 2013 at 12:09 pm**

I spent some time cracking the 2002 MINI Cooper 5 speed CAN protocol quite a while ago. Here's some more information: http://bobodyne.com/web-docs/robots/MINI/CAN/Presentation/index.html
A table of information I figured out. http://bobodyne.com/web-docs/robots/MINI/CAN/html/table.html
Raw data and small programs are here: http://bobodyne.com/web-docs/robots/MINI/CAN/

Reply Report comment

renaissanceman says:
October 23, 2013 at 8:25 am

Starting a 3 day course on CAN bus in december for my job, this is a great preparation for me.

Thanks !!

Reply Report comment

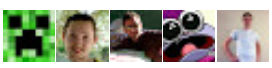
## Leave a Reply

Enter your comment here...



## NEVER MISS A HACK

HACKADAY on Follow


+20,937

Like Send 17,020 people like this. Be the first of your friends.