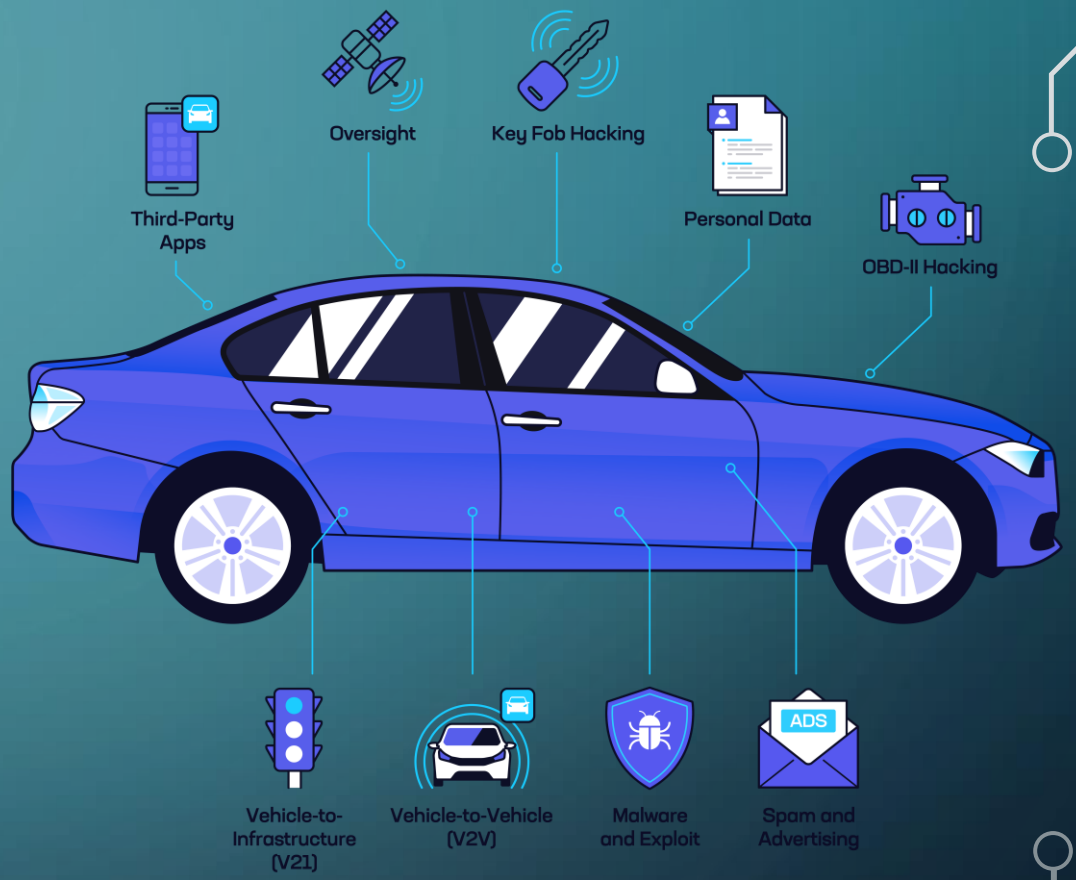
TOOL PER LA REPLICA DI DATASET DI FRAMES CAN SU SOCKET VIRTUALE RISPETTANDO TEMPISTICHE TEMPORALI

Laureando: Patrik Brighenti
Relatore: Prof. Luca Ferretti
Correlatore: Prof. Dario Stabili

AUTOMOTIVE 4.0

Il rapporto tra industria automobilistica e quarta rivoluzione industriale è destinato a modellare una serie di innovazioni legate ad una sempre maggiore interconnettività tra gli autoveicoli.

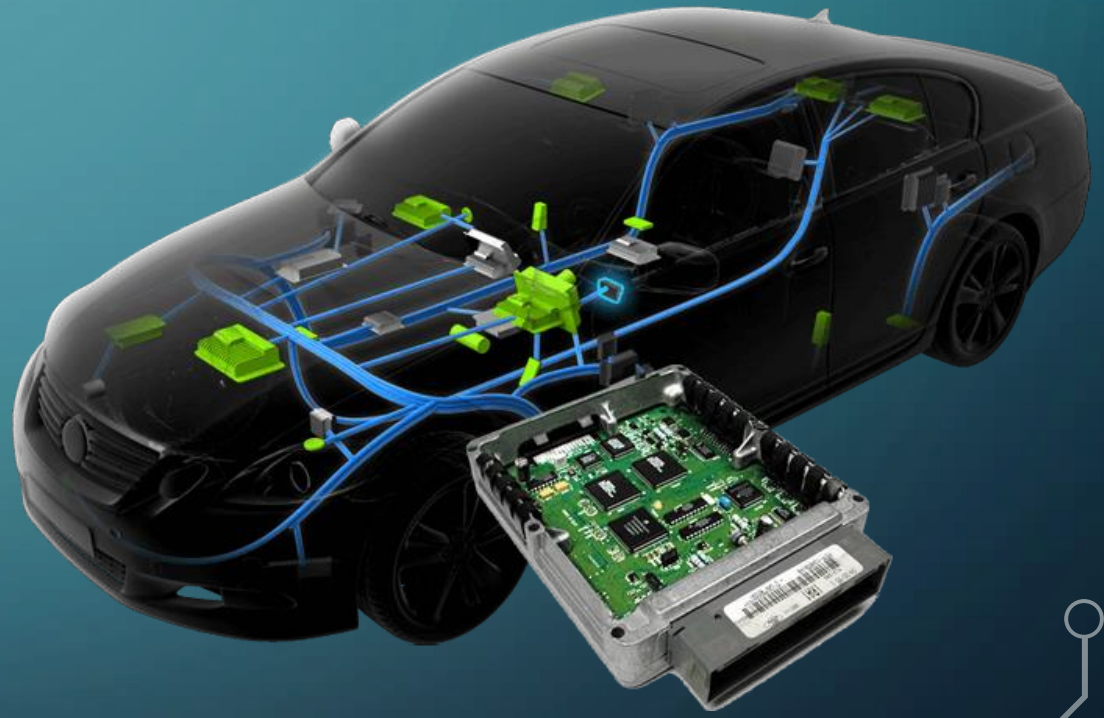
L'industria automotive ha preso coscienza solo recentemente del fatto che ci fosse un problema legato alla sicurezza. Passato l'entusiasmo di voler inserire sempre più features all'interno dei veicoli ci si è resi conto, infatti, che il tema della cyber security era di fondamentale importanza, specialmente per quei settori che stanno attraversando una fase di trasformazione digitale importante.



OBIETTIVO DELLA TESI

Al fine di fornire ai ricercatori un tool in grado di aiutarli a studiare i possibili attacchi ed eventuali possibili strategie difensive, si è deciso di implementare un software in grado di replicare dataset su una socket virtuale CAN riducendo al minimo il ritardo di inter-arrivo tra un frame e il successivo.

Il progetto è stato sviluppato utilizzando solo tools e risorse open source



PROTOCOLLO CAN

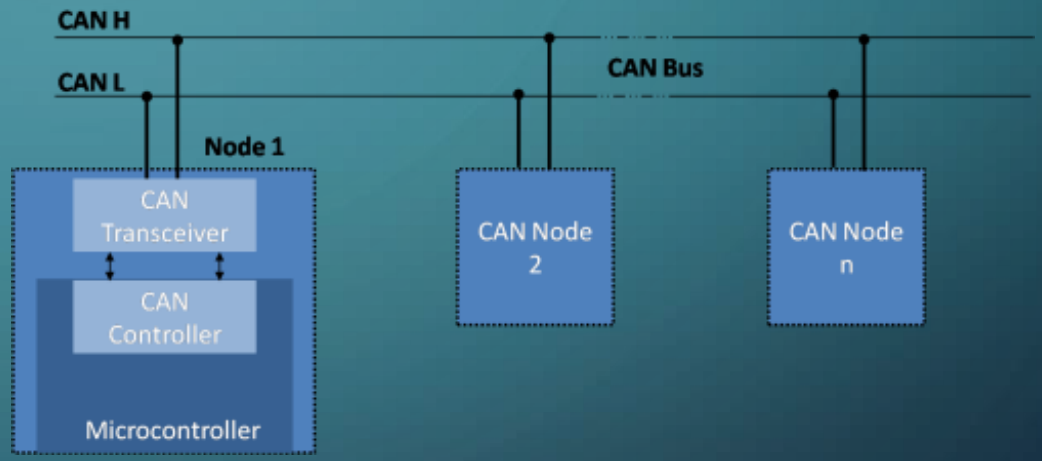
Il protocollo CAN è uno standard progettato per permettere ai microcontrollori all'interno della rete del veicolo di comunicare senza l'ausilio di un host computer.

Vantaggi:

- Affidabilità e Robustezza
- Economicità
- Flessibilità
- Velocità

Svantaggi:

Essendo un protocollo introdotto negli anni 80, la principale problematica è legata all'assenza di protocolli di sicurezza.



PROGETTO

1 – Specifiche Hardware
CPU AMD Ryzen 7 4800H dotata di 8 core fisici e 16 thread
RAM DDR4 16GB 3200 MHz

2 – Specifiche Software
Distro Linux installa su un'architettura a 64bit

3 – CAN-UTILS
Utility a riga di comando che contiene gli strumenti di base in grado di visualizzare, registrare, generare e riprodurre il traffico CAN su di una socket fisica o virtuale

4 – Linguaggio C – Modifica del programma Can-player

5 – Linguaggio Python – Scrittura degli script:
1) generateTestFile.py
2) main.py
3) analizeReport.py

CAN-UTILS

```
pi@raspberrypi:~ $ candump can0 can1
  can0  123   [8]  10 3E 48 65 6C 6C 6F 2C
  can1  123   [8]  10 3E 48 65 6C 6C 6F 2C
  can0  456   [3]  30 0A 05
  can1  456   [3]  30 0A 05
  can0  123   [8]  21 20 74 68 69 73 20 69
  can1  123   [8]  21 20 74 68 69 73 20 69
  can0  123   [8]  22 73 20 61 20 6C 6F 6E
  can1  123   [8]  22 73 20 61 20 6C 6F 6E
  can0  123   [8]  23 67 20 70 61 79 6C 6F
  can1  123   [8]  23 67 20 70 61 79 6C 6F
  can0  123   [8]  24 61 64 20 73 65 6E 74
```

- Ai fini della realizzazione del progetto sono stati utilizzati 2 tools presenti nell'utily CAN-UTILS:

  - Canplayer: permette di riprodurre un dataset su di una socket CAN. Dal suo codice sorgente è stata implementata una modifica per ottenere una maggiore precisione in termini d prestazioni temporali.

  - Candump: permette di loggare su files i frames in transito su di una socket CAN

PROGRAMMAZIONE REAL-TIME

1 – Inizializzazione delle strutture dati

2 – Minimizzare il numero di operazioni che intercorrono tra l'invio di un frame e il successivo

3 – Margine di anticipo euristico prima dell'invio di un frame

4 – Active sleep

5 – Stabilizzazione dello stato della scoket

DATASET

Il dataset è stato ottenuto dalla seguente risorsa:
https://sites.google.com/a/hksecurity.net/ocslab/Datasets/CAN-intrusion-dataset

- Frames rappresentati con ID, DLC e Payload
- 51 ID diversi
- 7 diverse tracce
- 8 milioni di messaggi corrispondenti a circa 90 minuti di traffico sulla rete
- Raccolti da diverse sessioni di guida su differenti tipologie di strade (urbane, extraurbane, ecc…)
- Differente viabilità delle strade
- Differenti condizioni metereologiche
- Differenti aree geografiche (pianura, collina e montagna)

ANALISI DEI RISULTATI

Valore	Occorrenze	Percentuale Occorrenze	Distribuzione normale
-2	54	0,006707908	0,023690592
-1	30913	3,840028819	0,150931451
0	494242	61,3949964	0,363506772
1	260012	32,29882487	0,330958993
2	6448	0,800973889	0,11391077
3	2738	0,340115774	0,014821232
4	2168	0,269310079	0,000729009
5	1868	0,232043924	1,35553E-05
6	1531	0,19018161	9,52832E-08
7	1284	0,159499143	2,53193E-10
8	1058	0,131425306	2,54341E-13
9	798	0,099127972	9,65848E-17
10	588	0,073041664	1,38653E-20
11	420	0,052172617	7,52457E-25
12	266	0,033042657	1,54369E-29
13	203	0,025216765	1,19721E-34
14	164	0,020372165	3,51001E-40
15	121	0,015030682	3,89022E-46
16	1	0,000124221	1,62993E-52
17	82	0,010186082	2,58163E-59
18	61	0,007577451	1,54578E-66

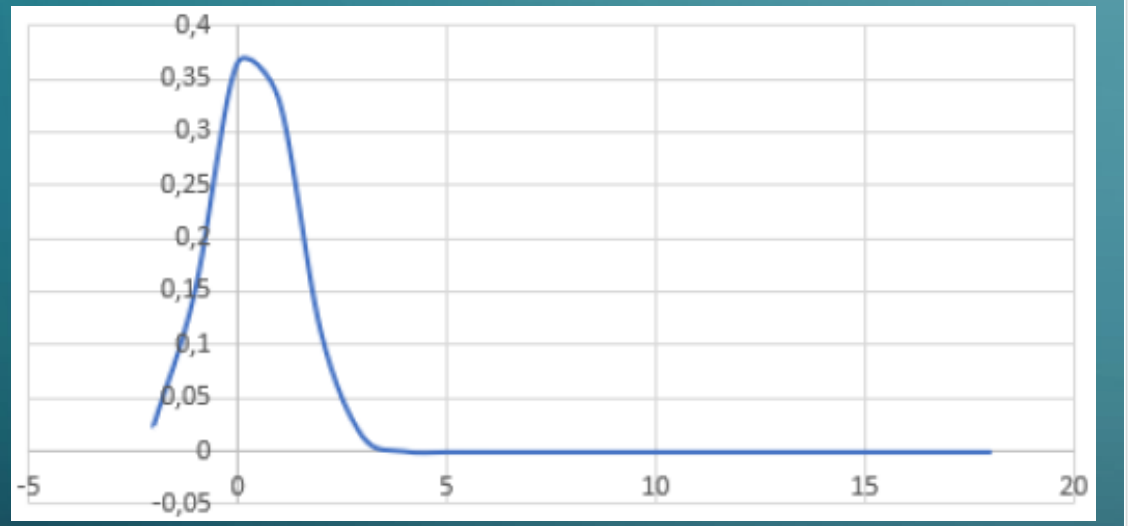
CAMPIONE IN ESAME:

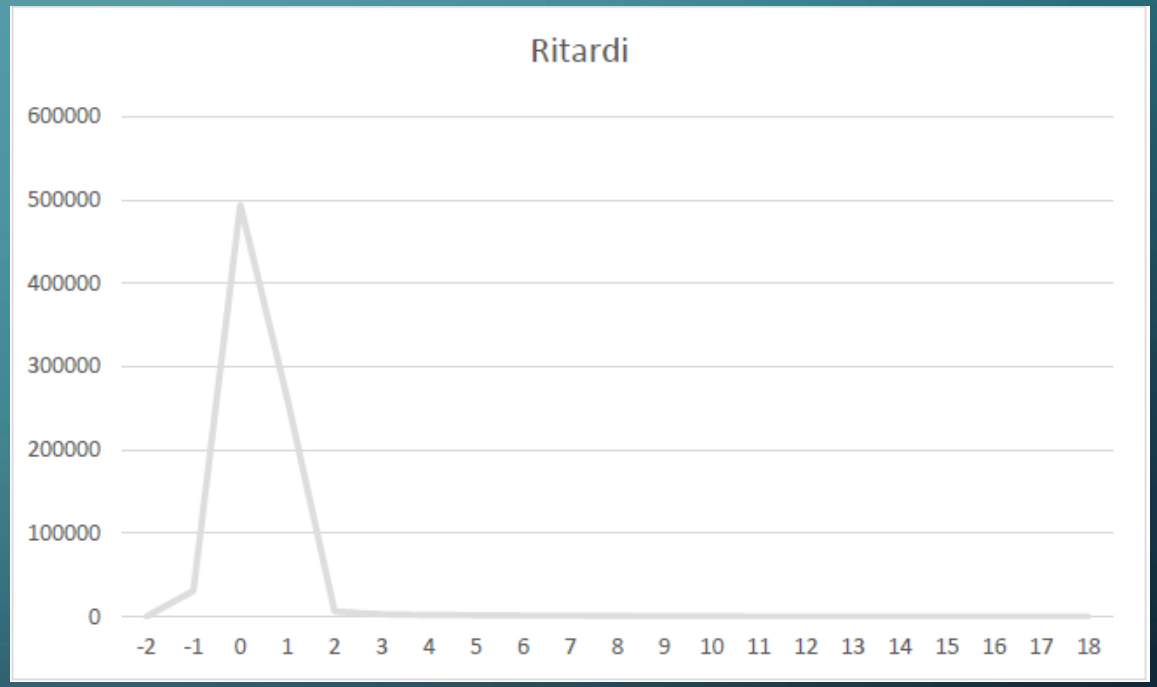
- 805599 frames inviati
- 254 simulazioni da dataset differenti
- Ritardo medio dell'invio di un frames: 0.431 microsecondi
- Deviazione standard: 1.013896

STATO DELLE RISORSE ALL'ESECUZIONE:

- 417 processi in esecuzione
- 3,48 GB di RAM utilizzata su 16 GB

ANALISI DEI RISULTATI





CONCLUSIONI

Il tool sviluppato riesce ad inviare oltre il 95% dei frames con un ritardo compreso nell'intervallo [-1,1] microsecondi.

Essendo eseguito su di un sistema operativo general purpose non abbiamo un comportamento deterministico ma statisticamente rilevante.

Il software è stato realizzato prevedendo la definizione di parametri che permettano di adattarsi a differenti comportamenti del sistema operativo su cui è in esecuzione.

Considerando la natura open source del progetto, lo sviluppo rimane aperto ad eventuali nuove features e migliorie.

GRAZIE A TUTTI PER L'ATTENZIONE

TOOL PER LA REPLICA DI DATASET DI FRAMES CAN SU SOCKET VIRTUALE RISPETTANDO TEMPISTICHE TEMPORALI

Studente: Patrik Brighenti
Relatore: Prof. Luca Ferretti
Correlatore: Prof. Dario Stabili