Mert D. Pesé, Jay W. Schauer, Junhui Li,
and Kang G. Shin

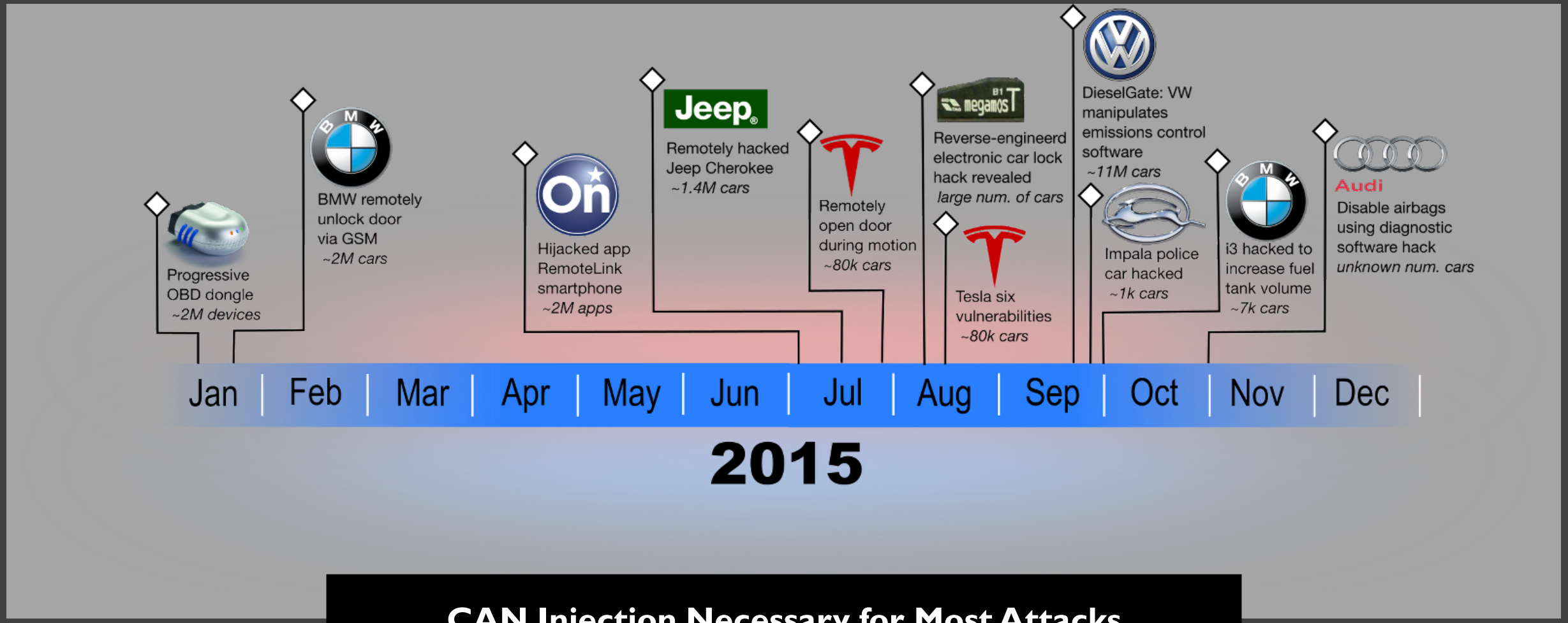
S2-CAN: Sufficiently Secure Controller Area Network

ACSAC 2021, Virtual
12/09/21

CSE COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

REAL-TIME COMPUTING LABORATORY
@ UNIVERSITY OF MICHIGAN, ANN ARBOR

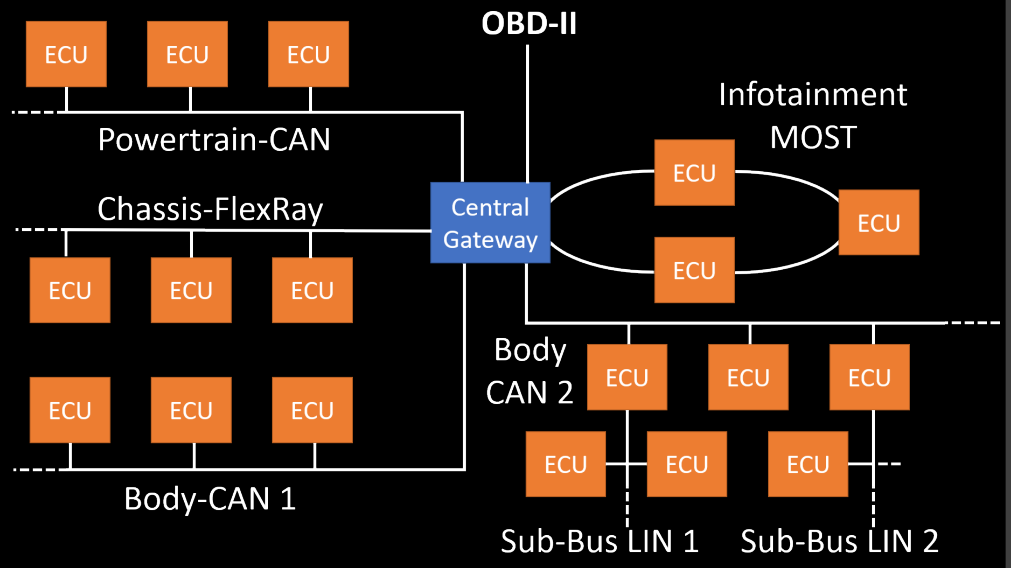
Most car hackings have one thing in common!



CAN Injection Necessary for Most Attacks

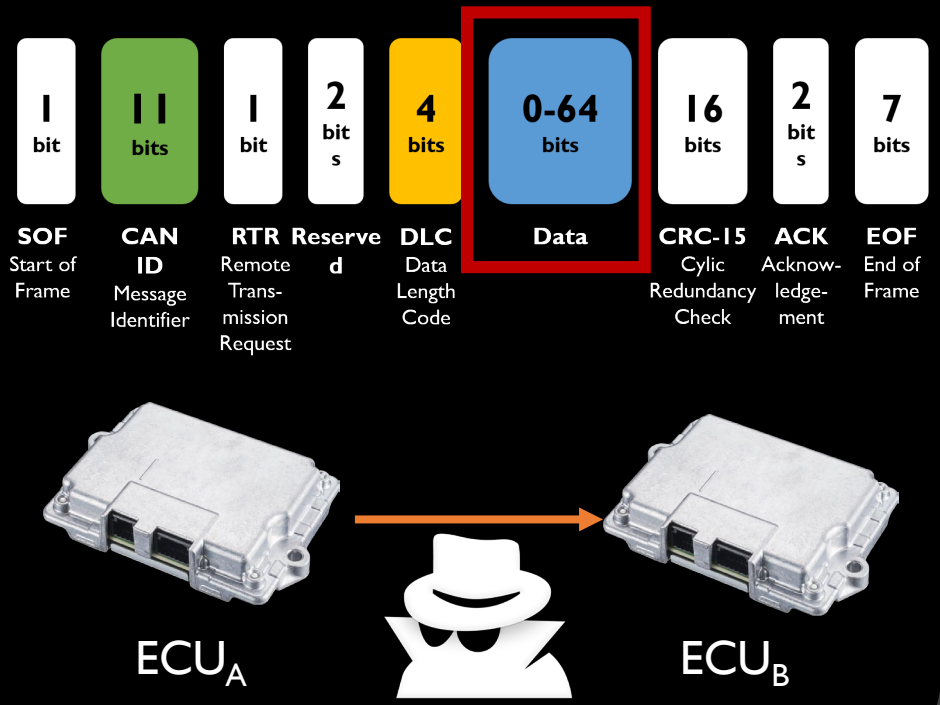
CAN Injection?!

In-Vehicle Network Architecture



CAN bus is most prominent & de facto standard

Controller Area Network



Lacks confidentiality and authenticity

Why is encryption required?

CAN Injection?!

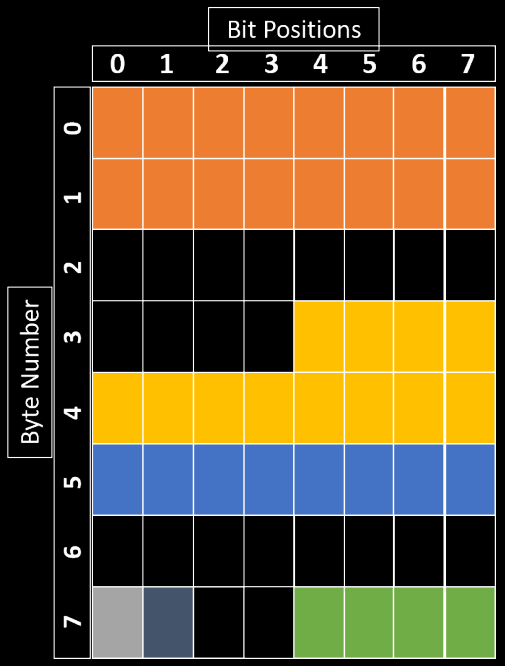
OBJECTIVE

Inject Well-Formed CAN Message to IVN

GOAL

Compromise or Break Vehicle's Functionalities

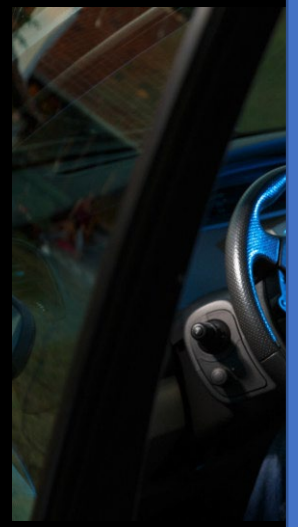
CHALLENGE



Semantics/Translation Tables Proprietary to OEM

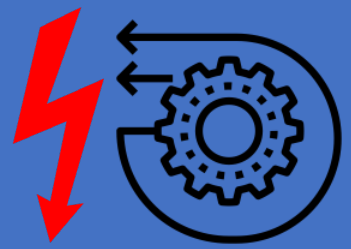
Encryption prevents reverse engineering!

CAN Injection?!



"Security by obscurity"

Current Automotive Standard



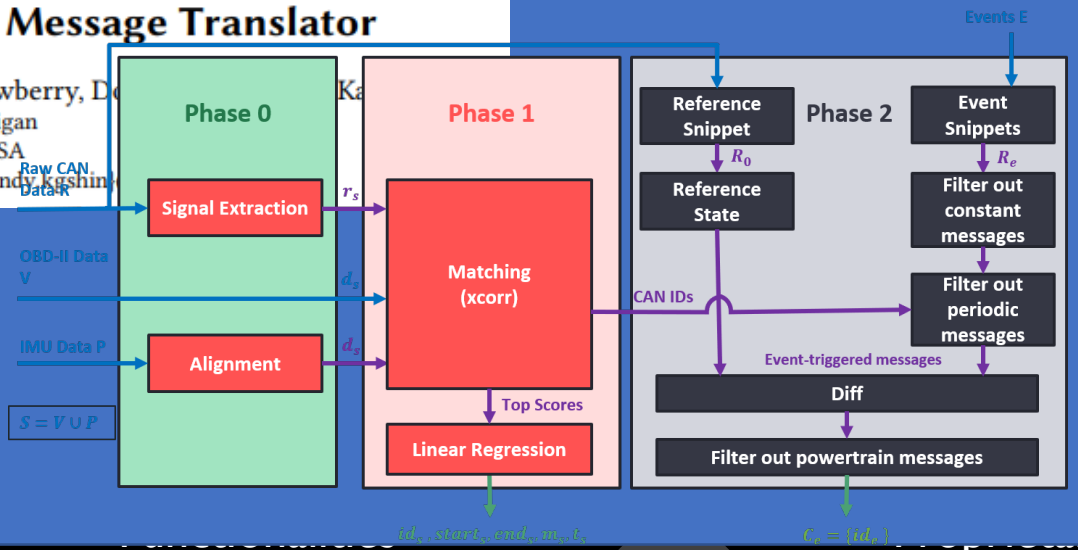
Automated CAN Bus Reverse Engineering

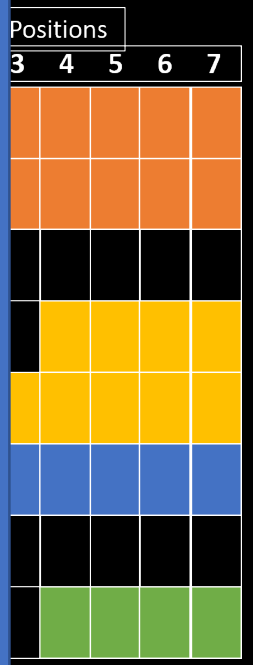
LibreCAN: Automated CAN Message Translator

Mert D. Pesé, Troy Stacer, C. Andrés Campos, Eric Newberry, D… Ka…
University of Michigan
Ann Arbor, MI, USA
{mpese,trstacer,andcmps,emnewber,chendy,kgshin}

[CCS'19]

- High accuracy
- Large coverage
- Fast (<2 min)





Inject Well-

…slation Tables

…y to OEM

Encryption prevents reverse engineering!

So, what‘re out there?

	Protection	Algorithm	HW/SW	Bus Load	Latency	MAC Length	Security Level
CaCAN [Ku14]	Authenticity + Freshness	SHA256-HMAC	HW+SW	+100%	+2.2-3.2µs	1 Byte	2^7
IA-CAN [Ha15]	Authenticity	Random. CAN ID + CMAC	SW	+0%	8bit: +72ms 32bit: +150µs	1-4 Bytes	2^7 - 2^{31}
vatiCAN [Nü16]	Authenticity + Freshness	SHA3-HMAC	SW	+16.2%	+3.3ms	8 Bytes	2^{63}
TESLA [Pe00]	Authenticity + Freshness	PRF + HMAC	SW	+0%	+500ms	10 Bytes	2^{79}
LeiA [Ra16]	Authenticity + Freshness	MAC	SW	+100%	N/A	8 Bytes	2^{63}
CANAuth[He11]	Authenticity + Freshness	HMAC	HW+SW	+0%	N/A	10 Bytes	2^{79}

COST

Resource-constrained (legacy) ECUs

LATENCY

Hard Real-Time Requirements

BUS LOAD

Must be below 80%, ideally below 30% to avoid scheduling issues

So, what‘re out there?

Protection	Algorithm	HW/SW	Bus Load	Latency	MAC Length	Security Level
S2-CAN	Confidentiality + Authenticity + Freshness	Circular Shift + Internal ID Match	SW	+0%	+75µs	N/A	$\sim 2^{49}$

S2-CAN: Our SOLUTION

- Breaks away from traditional cryptography-based solutions (S-CAN)
- Addresses three key feasibility issues
- Offers good practical -- albeit relaxed -- security guarantees

TRADE-OFF BETWEEN PERFORMANCE AND SECURITY

COST
Resource-constrained (legacy) ECUs

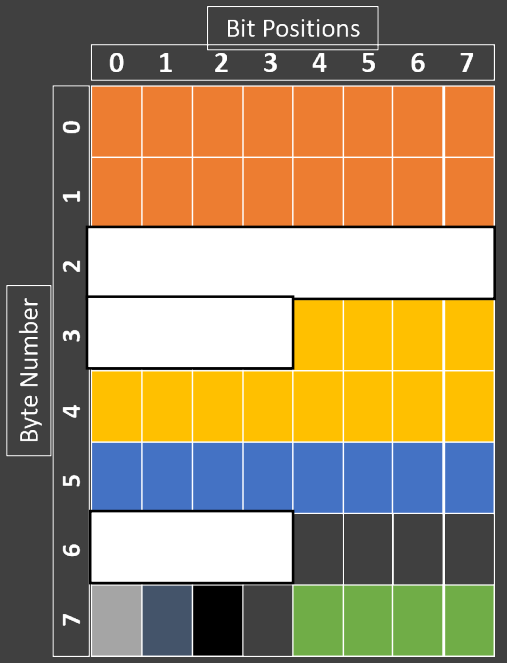
LATENCY
Hard Real-Time Requirements

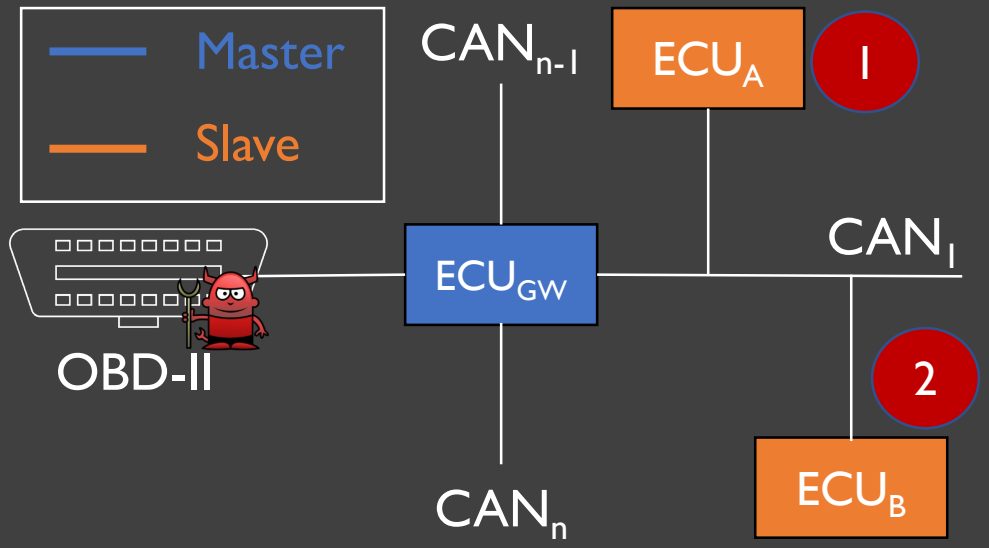
BUS LOAD
Must be below 80%, ideally below 30% to avoid scheduling issues

How does S2-CAN work?

(1) Internal ID = Rand(0, N-1)
(2) Internal Position = FS(𝕐)
(3) Internal Counter = Rand(0, 2^{16}-1)

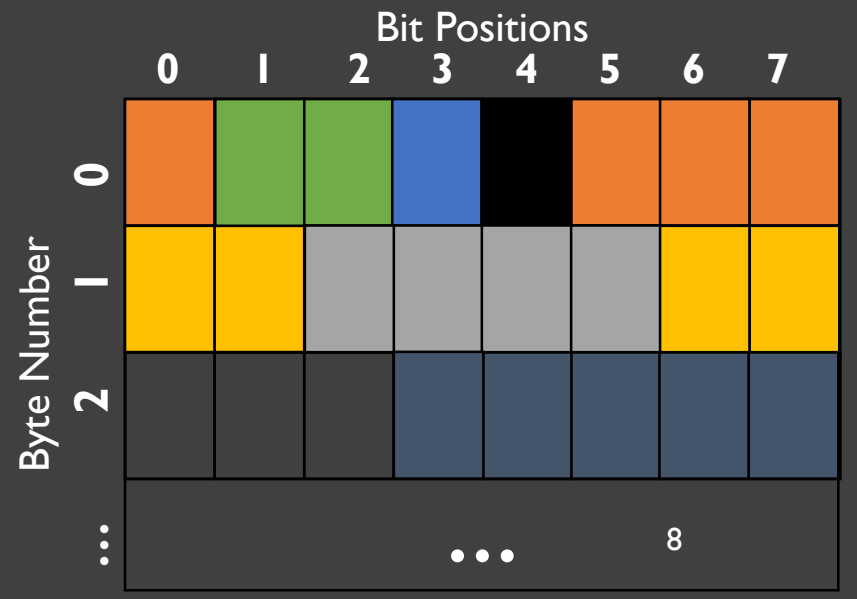
q_j = LEFTZEROPAD(int_ID_j , 8) ⊕ cnt_j





(4) Encoding Parameter f = (r_0, r_1, r_2, r_3, r_4, r_5, r_6, r_7), $r_i \in [0,7]$

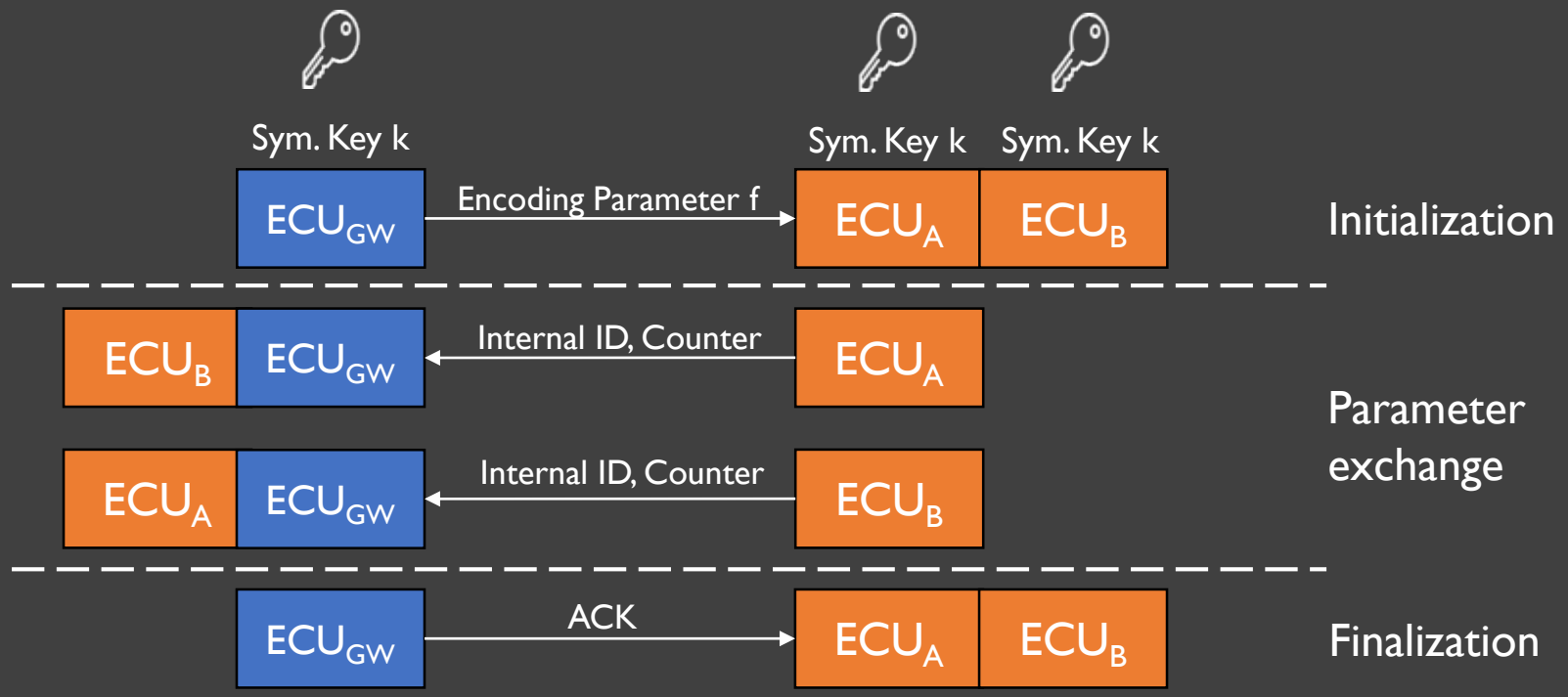
Circular Shift
Example: f = (3, 2, 1,...)



How to obtain these 4 parameters?

PERIODIC HANDSHAKES

S2-CAN Handshake



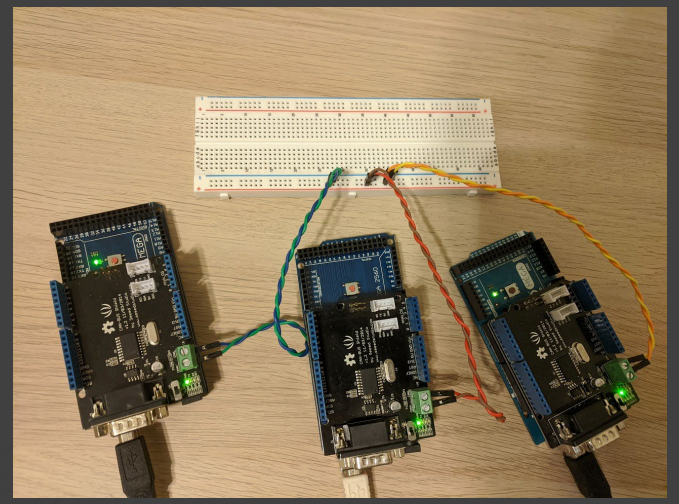
Security Requirements

All traffic encrypted with AES128 and authenticated with HMAC SHA256.

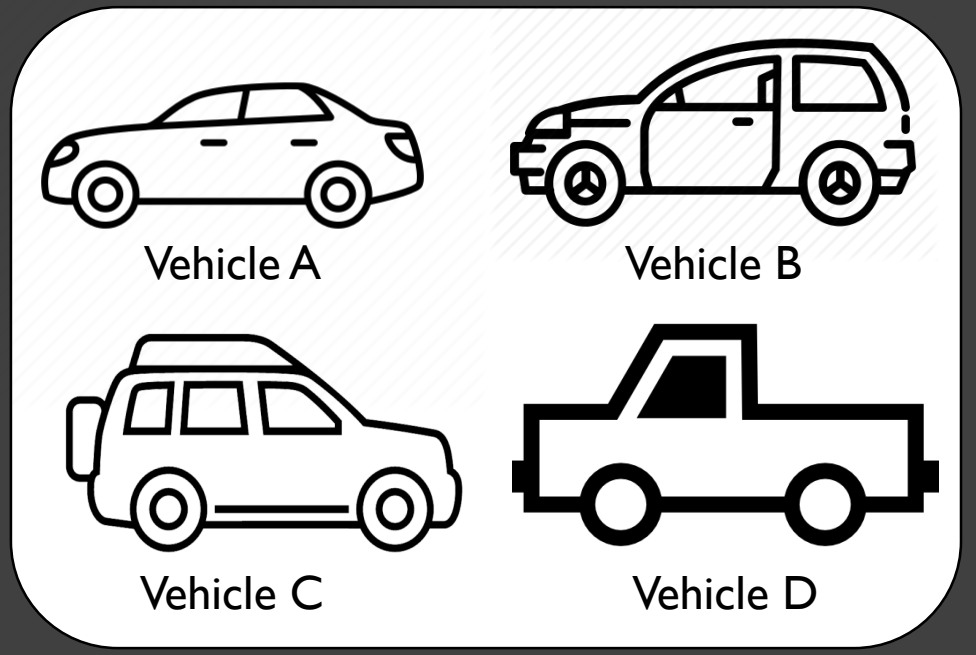
Handshake is periodic, needs to be repeated for every new session with session cycle *T*.

Experimental Setup

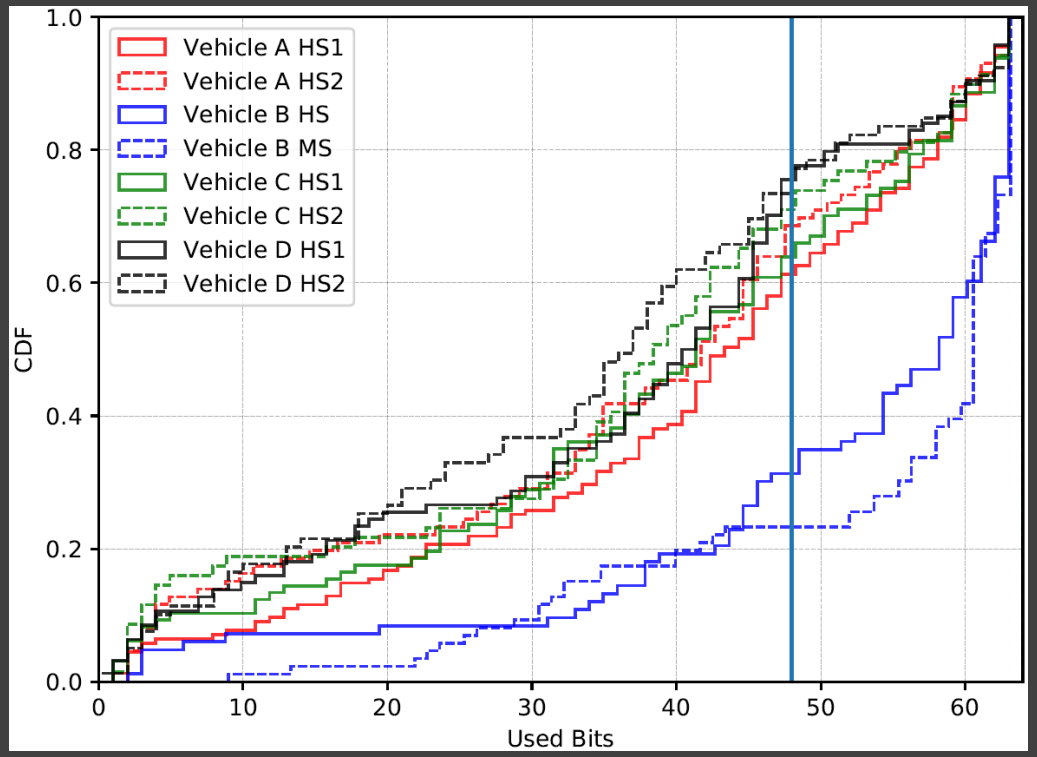
- Benchmark of latency and computational resources
  - 3x Arduino Mega 2560 with SeeedStudio CAN Shield



- Free Space and Security Analysis
  - Four different models of same OEM
  - Ground truth translation tables (“DBC files”) for Free Space analysis
  - Raw CAN Data collected with OpenXC, applied S2-CAN for security analysis



Evaluation: Free Space in CAN IDs



Vehicle	Bus	#CAN IDs	#Rebalan-cable IDs	#IDs with Free Space	Usable CAN IDs (%)
Vehicle A	HS1	102	31	63	**92.2**
	HS2	53	2	35	69.8
Vehicle B	HS	81	5	26	**38.3**
	MS	62	3	16	30.6
Vehicle C	HS1	57	7	38	**78.9**
	HS2	42	1	26	64.3
Vehicle D	HS1	58	7	43	**86.2**
	HS2	51	4	38	82.4

- **60-80% of all CAN IDs can be used by default**
- **Re-balancing further helps increase # of usable CAN IDs**

Evaluation: Benchmark

Handshake Latency

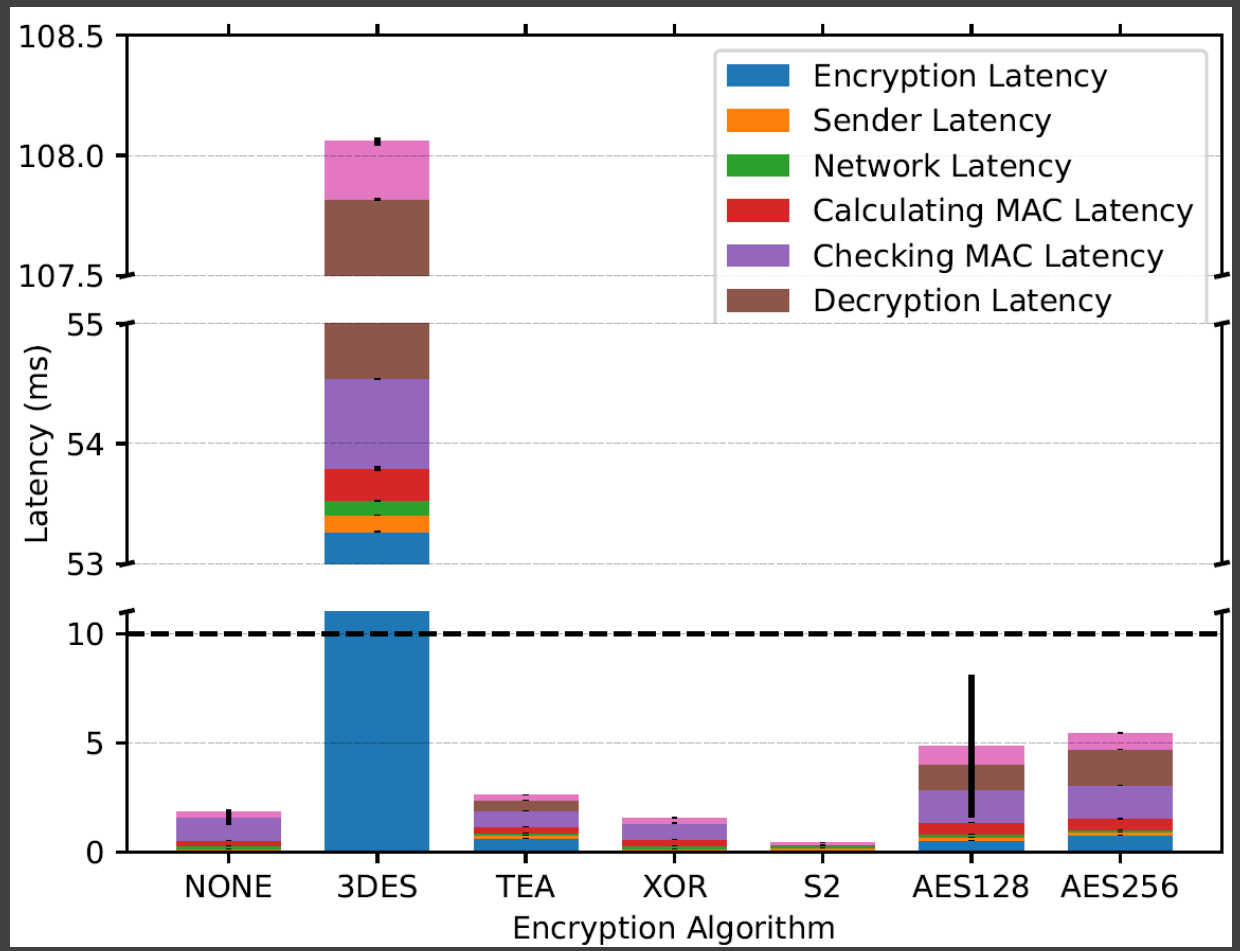
# Slave ECUs	2	5	10	25
Avg. Total Handshake Time (ms)	303	529	907	2037

- Handshake finishes in 2 seconds after starting the car
- New handshake overlaps with previous session, no "black-out"

Other Metrics

- CPU Overhead: 0.04%
- RAM Overhead: 0.8%
- Flash Memory Overhead: 1.3%

Operation Latency



- S2-CAN has overhead of 75 µs
- 44x faster than next-best approach

Evaluation: Security Analysis

- Brute-force integrity parameters: ~ 2^{49} combinations
- "Smart Attack" with LibreCAN+

1. Crack Encoding: 400 combinations
2. Authenticate Correctly: Determine counter position and internal ID

$$t_a = t_r + t_{st1} + t_{st2} + t_i \approx t_r + t_{st1} > T$$

S2-CAN secure if Session Cycle T smaller than total attack time!

t_{st1}	CAN (LibreCAN)	S2-CAN (LibreCAN+)
Veh. A	0:27	10:33
Veh. B	0:36	18:32
Veh. C	0:26	10:42
Veh. D	0:26	10:52

Trace Length (%)		5	10	25	50	75	100
Veh. A	ST1	11/20	6/10	4/4	3/3	2/2	1/1
	ID	10/20	6/10	4/4	3/3	2/2	1/1
	cnt	11/20	6/10	4/4	3/3	2/2	1/1
Veh. B	ST1	12/20	4/10	3/4	2/3	1/2	1/1
	ID	11/20	3/10	3/4	1/3	1/2	1/1
	cnt	12/20	4/10	3/4	2/3	1/2	1/1
Veh. C	ST1	8/20	5/10	3/4	3/3	2/2	1/1
	ID	8/20	5/10	3/4	3/3	2/2	1/1
	cnt	8/20	5/10	3/4	3/3	2/2	1/1
Veh. D	ST1	6/20	3/10	0/4	0/3	0/2	0/1
	ID	6/20	3/10	0/4	0/3	0/2	0/1
	cnt	6/20	3/10	0/4	0/3	0/2	0/1

$t_{r,min}$

¼ of t_{st1}

T_{max} ≈ 18-20 minutes

Conclusion

Secure and Feasible CAN Bus Possible by Security-Performance Tradeoff

Feasibility

First secure CAN approach to satisfy OEM requirements, guaranteed backward-compatible

Performance

Negligible resource overhead compared to regular CAN

Security

Secure with correct choice of session cycle

Q & A

Mert D. Pesé

Jay W. Schauer

Junhui Li

Kang G. Shin

References

[Ku14] Kurachi, R., Matsubara, Y., Takada, H., Adachi, N., Miyashita, Y. and Horihata, S., 2014, November. CaCAN-centralized authentication system in CAN (controller area network). In 14th Int. Conf. on Embedded Security in Cars (ESCAR 2014).

[Ha15] Han, K., Weimerskirch, A. and Shin, K.G., 2015. A practical solution to achieve real-time performance in the automotive network by randomizing frame identifier. Proc. Eur. Embedded Secur. Cars (ESCAR), pp.13-29.

[Nü16] Nürnberger, S. and Rossow, C., 2016, August. –vatican–vetted, authenticated can bus. In International Conference on Cryptographic Hardware and Embedded Systems (pp. 106-124). Springer, Berlin, Heidelberg.

[Pe00] A. Perrig, R. Canetti, J. Tygar, and D. Song. 2000. Approaches for secure and efficient in-vehicle key management. In Proceedings of the IEEE Symposium on Security and Privacy (SP 2000).

[Ra16] A.-I. Radu and F.D. Garcia. 2016. LeiA: a lightweight authentication protocol for CAN. Askoxylakis, I., Ioannidis, S., Katsikas, S., Meadows, C. (eds.) ESORICS 2016 878 (2016).

[He11] A. Van Herrewege, D. Singelee, and I. Verbauwhede. 2011. CANAuth – a simple, back-ward compatible broadcast authentication protocol for CAN bus. ECRYPT Workshop on Lightweight Cryptography (2011).