

Mercedes-Benz
Tech Innovation



OpenSSL Providers in the Automotive Environment

Peter Schmidberger & Dragan Zuvic
Cross Technologies

Mercedes-Benz



Agenda _

Motivation

Car Security: Different **Security Requirements** require different solutions

Our Journey

We need an **Innersource Provider** supporting developers and suppliers

Conclusion

What we learned helps others as well

We are Mercedes-Benz Tech Innovation _



Daniel Geisel



Christine Luckert

2.500

Employees

(in Germany, July 2025)

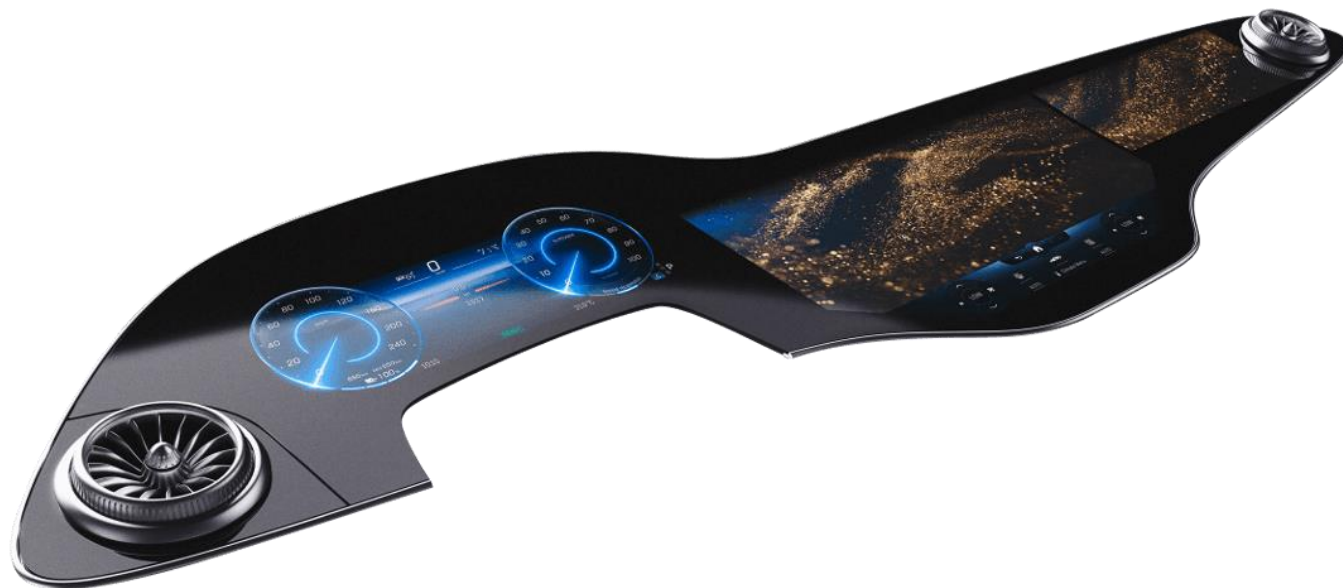
Where we are

Ulm (headquarters),
Stuttgart, Berlin, Karlsruhe

Worldwide Group network



We are a 100 % subsidiary of Mercedes-Benz,
but we don't build cars.



We are the tech partner of Mercedes-Benz
along the entire value chain - from product
development and manufacturing to sales,
mobility and after-sales services.

Motivation _

Car Security: Different **Security Requirements** require different solutions

Why Crypto? _

UNECE R 155 Regulation

- Introduces Cyber Security Management
- Reviewed by Higher federal authorities
- Leads to Cybersecurity Compliance & Certification

blackhat 2020 Hack:

- SkyGo worked > 1y before disclosure
- X-Ray of Chips / BGA soldering / "patching"
- Use less secure ECUs previous Gen

We invest in security **by law**

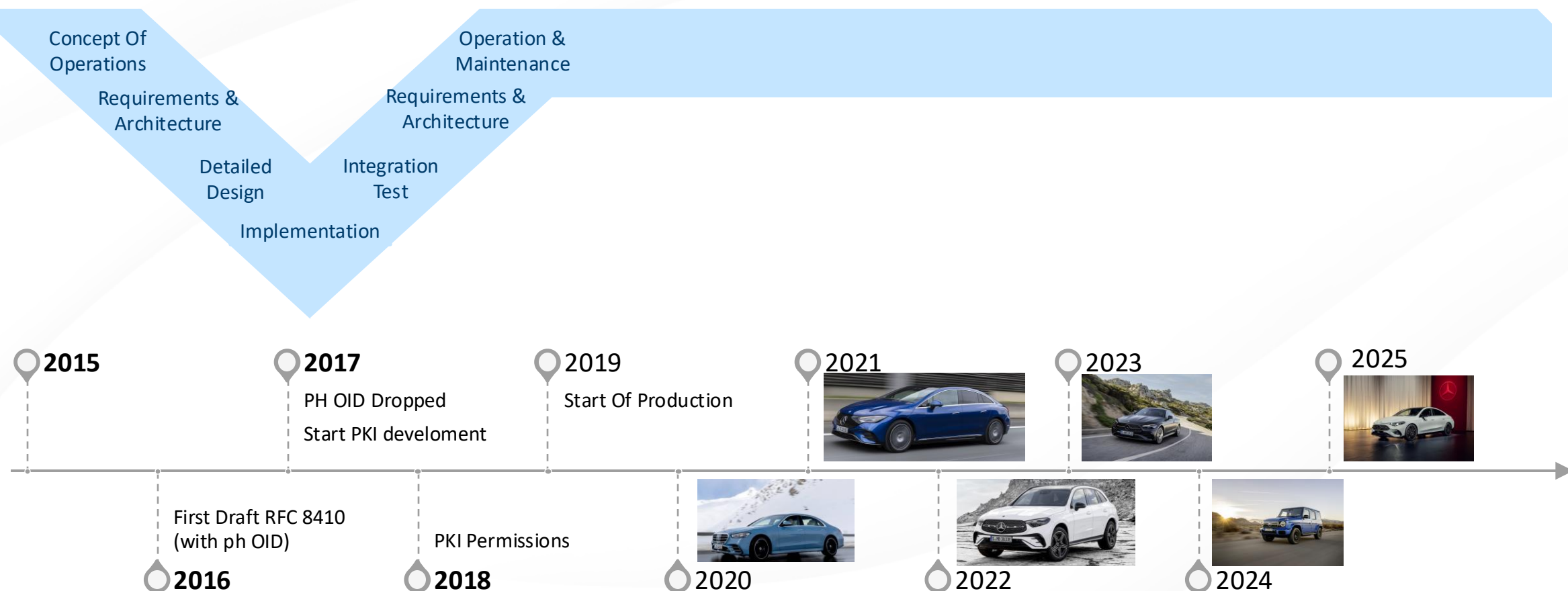
The bigger the **value**, the
bigger the **invest**

Why do we operate the PKI? _

- Ensuring the integrity of electronic control units (ECUs) in vehicles from the OEM (Mercedes-Benz)
- Access control to the ECUs by means of certificates (diagnostics, coding, ...)
- Positions of trust with other OEMs (e.g. mobile phone, charging infrastructure, ...)
- Communication to the vehicles



Why ed25519ph algorithm? _

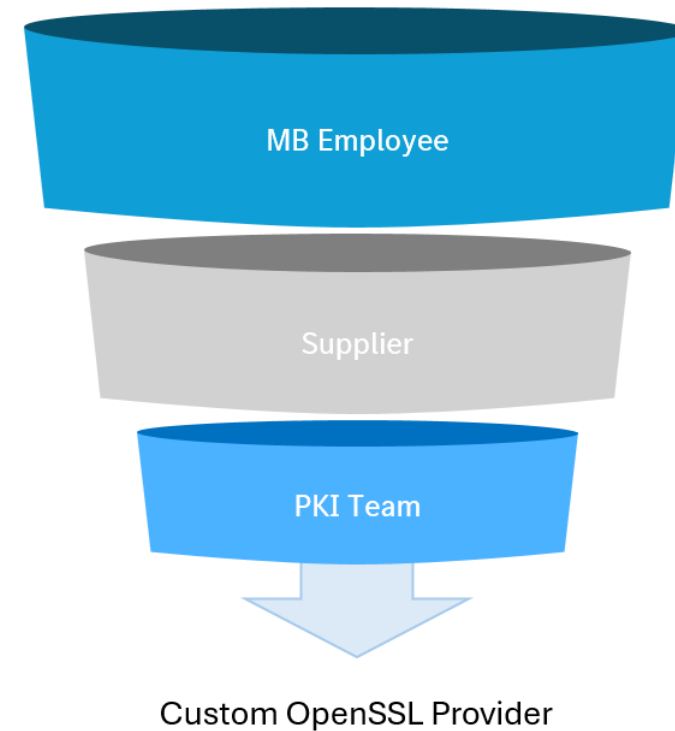


Vehicle development requires a long lead time and the service for our vehicles must be guaranteed for decades to come

Why we need OpenSSL? _

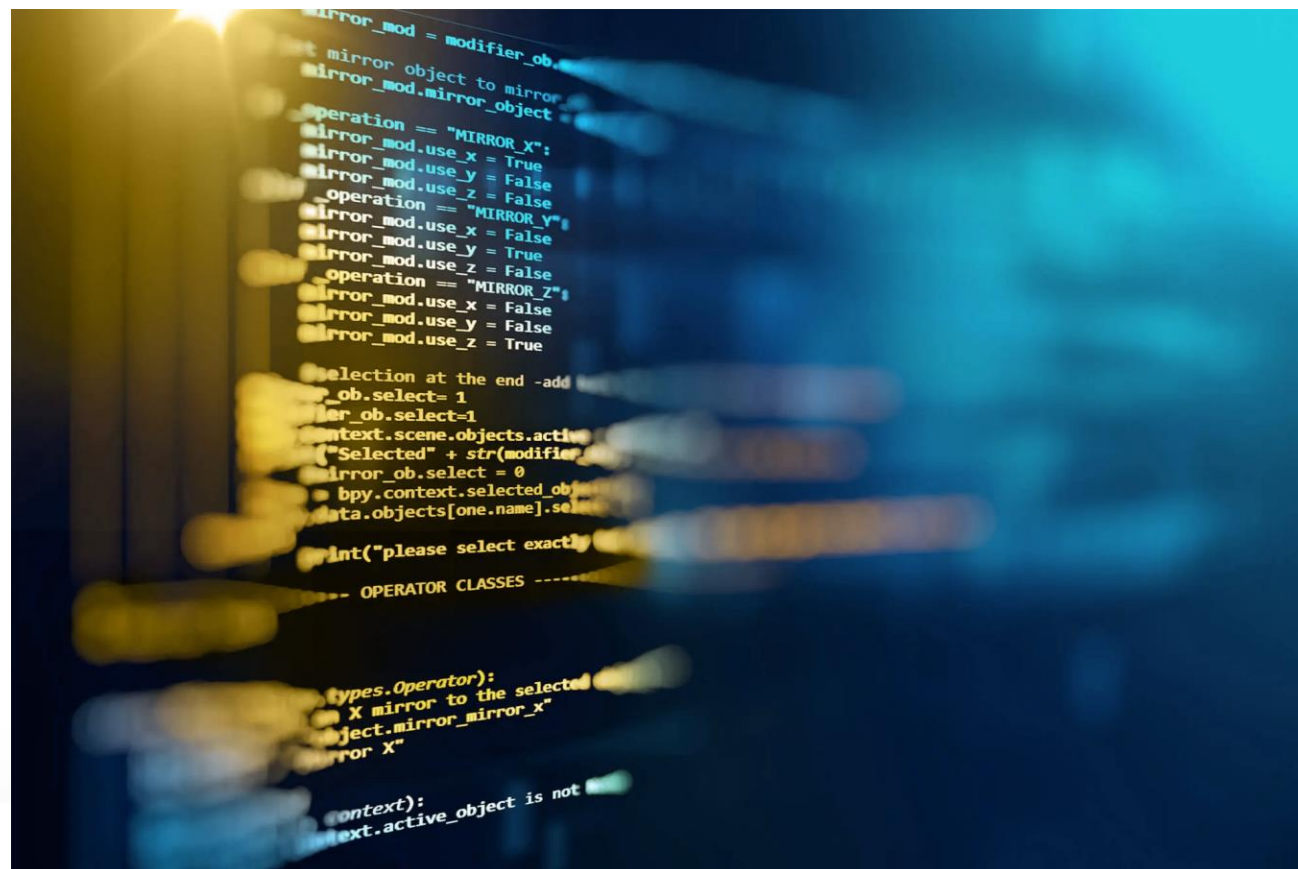
openssl is one of the most widely used freely available tools to perform crypto operations.

- Suppliers / ECU Developers:
Instead of using custom tools
- Mercedes-Benz Employee:
Able to verify certificates
- PKI Team:
Reference of the implementation



Features We Missed in OpenSSL _

- Support for ed25519ph
- X509 certificates with ed25519ph signatures
- Ability to use custom algorithm identifiers
- Support of attribute certificates
- Attribute certificates in PKCS#12 containers



Our Journey _

We need an **Innersource Provider**
supporting developers and suppliers

First Try _

- Find `ed25519` functions and add
 - Prehashed functions
- Should work out box

After one week not knowing, what we missed to implement:

We need external support!

```
prj/mbti/openssl-failed via v5.38.2
> ./util/opensslwrap.sh version
OpenSSL 3.1.0-dev (Library: OpenSSL 3.1.0-dev )

prj/mbti/openssl-failed via v5.38.2
> ./util/opensslwrap.sh list -signature-algorithms | grep
ED25519PH
{ 1.3.101.114, ED25519PH } @ default

prj/mbti/openssl-failed via v5.38.2
> ./util/opensslwrap.sh genpkey -algorithm ed25519ph -out
failed.pem
Error writing key
804BF49B3E7B0000:error:1D800065:ENCODER routines:OSSL_ENC
ODER_to_bio:reason(101):crypto/encode_decode/encoder_lib.
c:55:No encoders were found. For standard encoders you ne
ed at least one of the default or base providers availabl
e. Did you forget to load them?
804BF49B3E7B0000:error:04800073:PEM routines:do_pk8pkey:e
rror converting private key:crypto/pem/pem_pk8.c:133:

prj/mbti/openssl-failed via v5.38.2
> ./util/opensslwrap.sh genpkey -algorithm NOTEXIST -out
failed.pem
Error initializing NOTEXIST context
80DB285E78780000:error:0308010C:digital envelope routines
:inner_evp_generic_fetch:unsupported:crypto/evp/evp_fetch
.c:341:Global default library context, Algorithm (NOTEXIS
T : 0), Properties (<null>)

prj/mbti/openssl-failed via v5.38.2
> 
```

Provider vs. LFS switch _

```
struct ossl_dispatch_st {
    int function_id;
    void (*function)(void);
};

# define OSSL_DISPATCH_END
    { 0, NULL }
```

Function Pointer Tables OSSL_DISPATCH

OSSL queries operations

Reason: FIPS binary running with $\forall$ OSSL

```
struct inode_operations
{
    /* ... */

    int (*create) (struct mnt_idmap *, struct inode *,
        struct dentry *, umode_t, bool);
    int (*link) (struct dentry *, struct inode *,
        struct dentry *);
    int (*unlink) (struct inode *, struct dentry *);
    int (*symlink) (struct mnt_idmap *, struct inode *,
        struct dentry *, const char *);
    int (*mkdir) (struct mnt_idmap *, struct inode *,
        struct dentry *, umode_t);
    int (*rmdir) (struct inode *, struct dentry *);
    /* ... */
}
```

More OO Style: * f() in structs with types

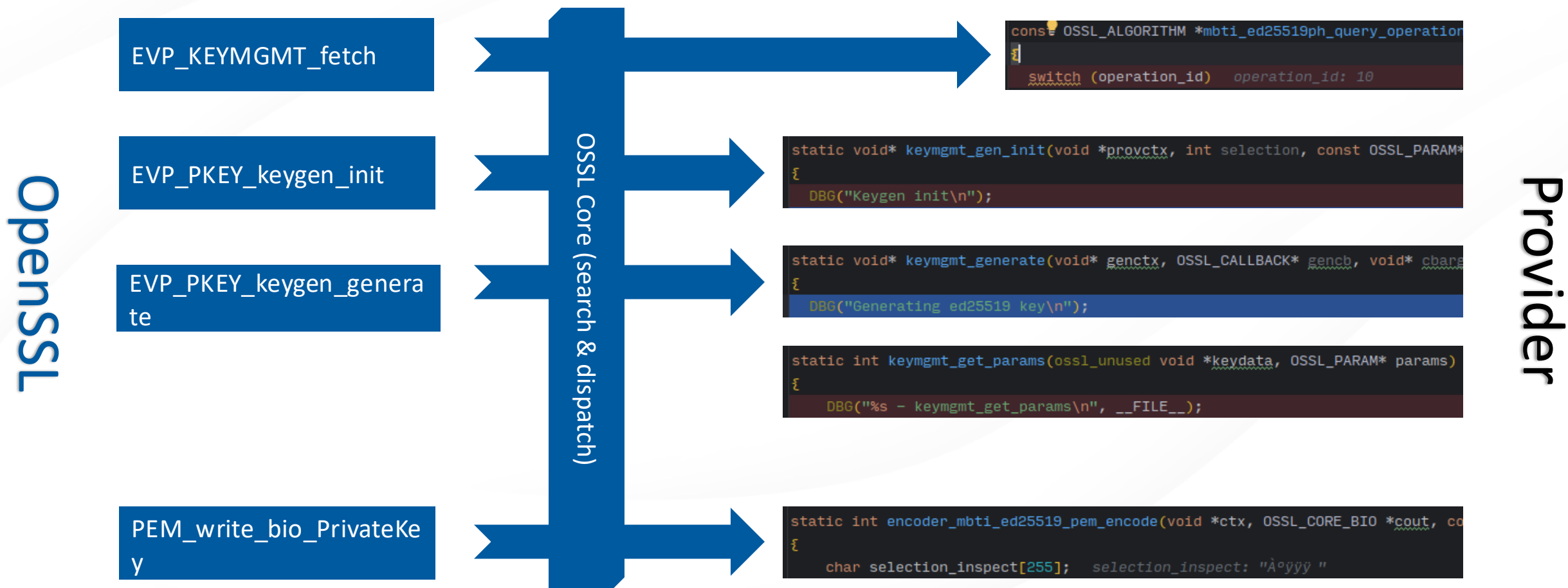
The driver constructs structures

Reason: File philosophy of Unix

Provider Use Cases _

	cipher	decoder	digest	encoder	kdf	kem	keyexch	keymgmt	mac	object	rand	signature	skeymgmt	storemgmt
Encryption with "-K"	•													
Hash			•											
Signature algorithm with x509		•		•				•				•		
oqs provider		•		•		•		•				•		
pkcs#11 provider	•	•	•	•	•		•	•		•	•	•	•	•

Real First Step: openssl genpkey _



Please sign here _

```
openssl pkeyutl -sign -inkey key.pem -in msg.raw -rawin -out signature.raw
```

OSSL_STORE_LOADER_*	Not implemented, but searched
OSSL_FUNC_DECODER_NEWCTX	Called 5x
OSSL_FUNC_DECODER_DECODE	BIO_new_from_core_bio d2i_PKCS8_PRIV_KEY_INFO_bio PKCS8_pkey_get0 EVP_PKEY_new_raw_private_key
OSSL_FUNC_KEYMGMT_LOAD	Called by call back function; creates provider side key
OSSL_FUNC_KEYMGMT_GET_PARAMS	Gets called to fill params with local param key/values: bits, security-bits, max-size, mandatory-digest
OSSL_FUNC_KEYMGMT_HAS	Checking the key
OSSL_FUNC_SIGNATURE_DIGEST_SIGN_*	Construct a key with EVP_PKEY_new_raw_private_key EVP_DigestSign with "provider=default" and "instance=ed25519ph"



For x509 we only needed _

`openssl req -new* -key k.pem -out tmp.csr`



50 x called

- Additionally, the following were required:
 - Register the OID / NID
 - ENCODER_ENCODE (der)

```

prj/mbti/ed25519ph-provider_tmp2 via v3.28.3
> openssl asn1parse -in root_csr.pem -i -dump
 0:d=0  hl=3  l= 193 cons: SEQUENCE
 3:d=1  hl=2  l= 109 cons: SEQUENCE
 5:d=2  hl=2  l=   1 prim: INTEGER           :00
 8:d=2  hl=2  l=  18 cons: SEQUENCE
10:d=3  hl=2  l=  16 cons: SET
12:d=4  hl=2  l=  14 cons: SEQUENCE
14:d=5  hl=2  l=   3 prim: OBJECT             :commonName
19:d=5  hl=2  l=   7 prim: UTF8STRING         :Root CA
28:d=2  hl=2  l=  48 cons: SEQUENCE
30:d=3  hl=2  l=  11 cons: SEQUENCE
32:d=4  hl=2  l=   9 prim: OBJECT             :1.3.6.1.4.1.2916.1337
43:d=3  hl=2  l=  33 prim: BIT STRING
0000 - 00 58 fe c4 c1 63 0a 15-33 af b1 30 db 0c 59 1c  .X...c...0..Y.
0010 - 65 33 18 0c 5d c9 85 5f-51 d0 d0 10 06 5a 64 3a  e3...]._Q....Zd:
0020 - a9
78:d=2  hl=2  l=  34 cons: cont [ 0 ]
80:d=3  hl=2  l=  32 cons: SEQUENCE
82:d=4  hl=2  l=   9 prim: OBJECT             :Extension Request
93:d=4  hl=2  l=  19 cons: SET
95:d=5  hl=2  l=  17 cons: SEQUENCE
97:d=6  hl=2  l=  15 cons: SEQUENCE
99:d=7  hl=2  l=   3 prim: OBJECT             :X509v3 Basic Constraints
104:d=7  hl=2  l=   1 prim: BOOLEAN           :255
107:d=7  hl=2  l=   5 prim: OCTET STRING
0000 - 30 03 01 01 ff
114:d=1  hl=2  l=  13 cons: SEQUENCE
116:d=2  hl=2  l=   9 prim: OBJECT             :1.3.6.1.4.1.2916.1337
127:d=2  hl=2  l=   0 prim: NULL
129:d=1  hl=2  l=  65 prim: BIT STRING
0000 - 00 8f 7c 1c de 68 7c 1d-ca 15 e1 d2 ee d0 a3 6f  ..|..h|.....o
0010 - e9 9e 9e dc ee cc 16 72-c7 34 b5 a6 25 82 79 15  .....r.4..%.y.
0020 - 47 38 53 d2 c7 09 67 68-3b 5c 92 a8 74 5b f0 9f  G8S...gh;\.t[.
0030 - 98 af a0 78 82 55 b9 d0-0c f4 1d ab cb 2d 25 92  ...x.U.....-%.
0040 - 06
prj/mbti/ed25519ph-provider_tmp2 via v3.28.3
> |

```

*Omitted options like `-subj "/CN=Root CA"` `--addext "basicConstraints=critical,CA:TRUE"`

Implemented functions _

decode

_newctx

_freectx

_decode

encode

newctx

freectx

encode (der)

encode (pem)

encode (txt)

keymgmt

new

free

Has

match

load

import

import_types

import_types_ex

gen_init

gen

gen_cleanup

gettable_params

get_params

provider

Init

Teardown

get_params

gettable_params

get_reason_strings

query_operation

signature

newctx

freectx

digest_sign_init

digest_sign

digest_sign_update

digest_sign_final

digest_verify_init

digest_verify_update

digest_verify_final

get_ctx_params

gettable_ctx_params

OpenSource ? _

Open for partners
in **shared projects**

InnerSource

**Mercedes-
Benz**

Supplier

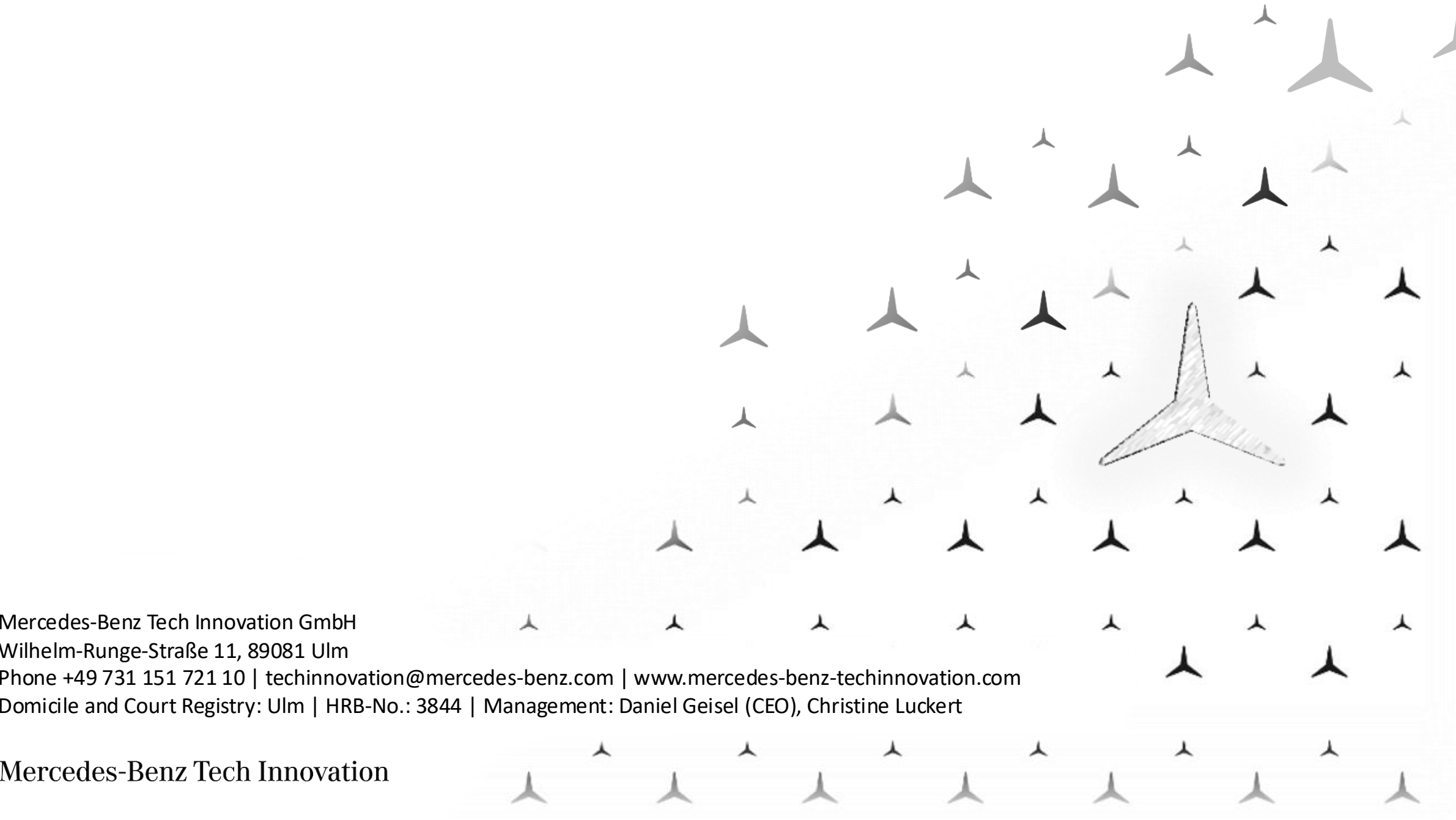
Conclusion _

What we learned helps others as well

In Retrospective _

- We **invested a lot** of effort for reaching only the minimum
- Most time we were **reading** documents or OpenSSL source code
- You **need a full-time developer** extending OpenSSL for your needs





Mercedes-Benz Tech Innovation GmbH

Wilhelm-Runge-Straße 11, 89081 Ulm

Phone +49 731 151 721 10 | techinnovation@mercedes-benz.com | www.mercedes-benz-techinnovation.com

Domicile and Court Registry: Ulm | HRB-No.: 3844 | Management: Daniel Geisel (CEO), Christine Luckert

Mercedes-Benz Tech Innovation

References _

KBA & UN / ECE

https://www.kba.de/DE/Themen/Typgenehmigung/Typgenehmigungserteilung/Cyber_SoftwareUpdate/Cyber_SoftwareUpdate_node.html

Security Research on Mercedes-Benz: From Hardware to Car Control

<https://i.blackhat.com/USA-20/Thursday/us-20-Yan-Security-Research-On-Mercedes-Benz-From-Hardware-To-Car-Control.pdf>

Linux VFS Documentation

<https://www.kernel.org/doc/html/v6.15/filesystems/vfs.html>

Inner Source License

https://opensource.mercedes-benz.com/news/sponsor_innersource_commonsoss/

Mercedes-Benz Cars:

<https://media.mercedes-benz.com/>