Exploiting hardware-backed keys with the EVP_SKEY API

OpenSSL Conference 2025, Prague

Holger Dengler (dengler@linux.ibm.com)
Reinhard Bündgen (buendgen@de.ibm.com)

IBM Deutschland R&D GmbH

Outline

Background

IBM Z specific

Exploitation

EVP_SKEY

Background

Hardware-backed Keys: Background

- clear key never readable in operating system/application
- key references/handles or wrapped keys
- implementation in separate hardware, firmware or CPU feature
- cryptographic operations: implicit clear key dereference or unwrap

IBM Z specific

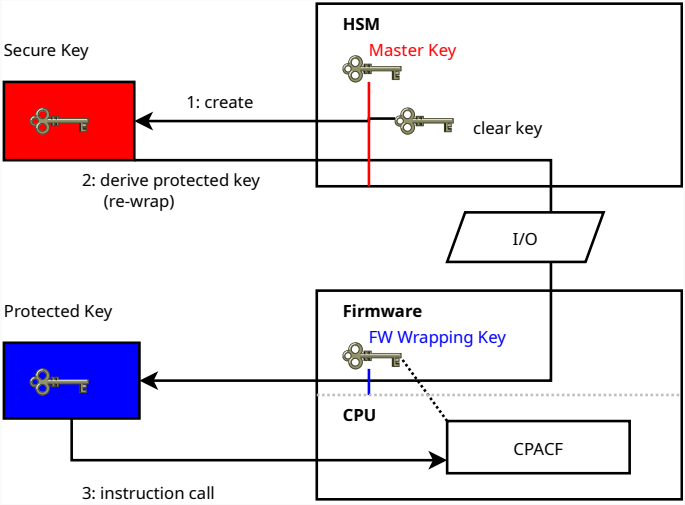
Cryptographic Acceleration on IBM Z

- Central Processor Assist for Cryptographic Function (CPACF)
  - instruction set for cryptographic operations
  - long running instructions, multiple cipher blocks per calls
  - instructions are interruptable and resumable
  - e.g. instruction KMC -> AES-128/192/256-CBC (w/ clear key or protected key)

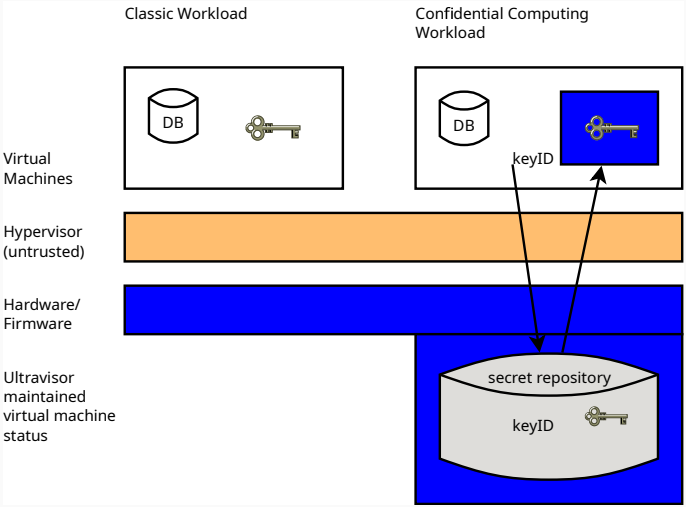
Hardware-backed Keys on IBM Z

- protected key
  - derived from key origins (persistent)
    - secure key (HSM)
    - retrievable secret ID (Confidential Computing)
    - clear key (Development/Test)
  - derive: rewrap with Firmware wrapping key (volatile)
    - FW wrapping key specific to virtual machine (partition, guest)
  - in main memory: only wrapped key
  - related clear key is never exposed to platform hypervisor, operating system or user-space
    - application: instruction KMC is called with the wrapped key (protected key)
    - firmware: implicit key un-wrap
    - firmware: perform AES-CBC algorithm

Hardware-backed Keys on IBM Z (Secure Key Example)



Hardware-backed Keys on IBM Z (Retrievable Secret Example)



Exploitation

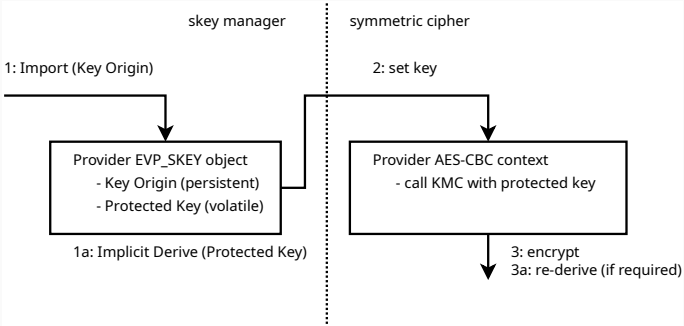
Protected Key exploitation today

- platform-specific open-source crypto library –> libzpc [1]
- pros:
  - minimal implementation
  - clean code base
- cons:
  - new, incompatible API
  - platform specific library
  - application requires modifications

Protected Key exploitation with OpenSSL Provider

- EVP SKEY API (since OpenSSL v3.5)
  - opaque key blob
  - provider defined key formats for symmetric operations
- EVP PKEY API
  - opaque key blob
  - provider defined key formats for asymmetric operations
- zpcprovider:
  - management for symmetric and asymmetric keys
  - ciphers for AES and ECDSA/EdDSA operations
  - future: decoder/encoder support
- pros:
  - common crypto API for applications
  - no platform-specific application code
- cons:
  - nothing found yet... ;)

Protected Key exploitation with OpenSSL (overview)



EVP_SKEY

Implementation: EVP_SKEY code sample

```
skey = EVP_SKEY_import_raw_key(..., key_origin, key_origin_len, ...);
...
ctx = EVP_CIPHER_CTX_new(...);
...
EVP_CipherInit_SKEY(ctx, ..., skey, ...);
...
EVP_CipherUpdate(ctx, ct, &ctlen, pt, ptlen);
...
EVP_CipherFinal_ex(ctx, ct, &ctlen);
...
```

Implementation: SKEY manager

- import/export: only persistent key origin
  - secure key (HSM)
  - retrievable secret ID (Confidential Computing)
  - clear key (Development/Test)
- export: no export to other providers possible
- protected key: implicitly derived from key origin
  - cached in EVP_SKEY object

Implementation: Symmetric Cipher

- cipher implementation similar to clear key
  - instruction configuration: parameter block
  - key object contains all required information to setup parameter block
  - one instruction call per operation
- differences
  - function code
  - additional re-derive handling

Future: Decoder/Encoder

- to-be-defined: file format for symmetric clear key
- to-be-defined: file format for symmetric hardware-backed keys
- PEM/DER: easy transition for applications: clear key -> protected key
- EVP_SKEY API requirements:
  - EVP_SKEY object ID
  - load API calls

Conclusion

- EVP_SKEY: similar flexibility as EVP_PKEY
- advantages of migration to EVP_SKEY
  - clear key works as of today
  - increase key protection level by using EVP_SKEY providers (e.g. zpcprovider or pkcs11-provider)
  - provide attributes to symmetric keys
- no platform-specific crypto API required for protected key exploitation

References

libzpc Repository. [Online]. Available: https://github.com/opencryptoki/libzpc.git

Exploiting hardware-backed keys with the EVP_SKEY API

OpenSSL Conference 2025, Prague

Holger Dengler (dengler@linux.ibm.com)
Reinhard Bündgen (buendgen@de.ibm.com)

IBM Deutschland R&D GmbH