

EVP_SKEY: symmetric keys beyond raw bytes

Opaque symmetric keys in OpenSSL

OpenSSL conference

October 7, 2025, Prague

Dmitry Belyavskiy, Principal Software Engineer, Red Hat

Who am I



Dmitry Belyavskiy

Red Hat Principal Software Engineer

Maintain: OpenSSL, OpenSSH

Current work: Post-Quantum transition in Red Hat

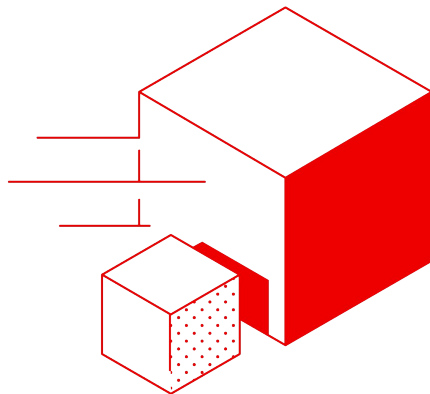
OpenSSL experience

Hacking since 2004

OpenSSL Technical Committee member since 2021

OpenSSL TACs/BAC member

Opaque symmetric keys – rationale



Hardware backed keys

PKCS#11, HSM, processor enclaves

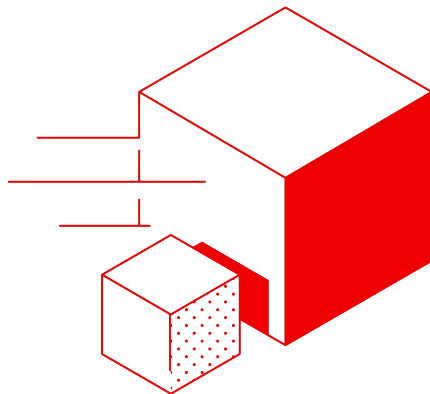
PKCS#11 provider

<https://github.com/latchset/pkcs11-provider>

Obfuscation/security

Key masking

Already implemented



Pre-3.5

Doable for EVP_CIPHER
Implementation-specific

OpenSSL 3.5

Support for symmetric ciphers and MACs

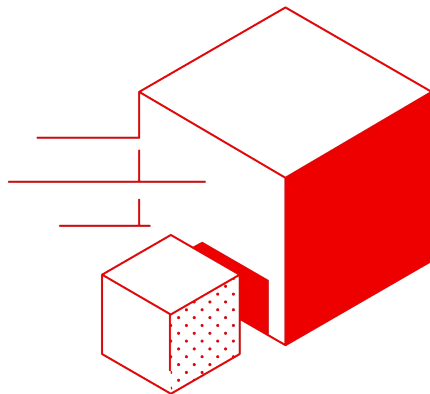
[PR #26753](#), [PR #26832](#)

OpenSSL 3.6

EVP_SKEY derivation from KDFs and key exchange

[PR #28369](#), [PR #28486](#)

Introducing EVP_SKEY structure



EVP_SKEY structure

```
struct evp_key_st {  
    /* == Common attributes == */  
    CRYPTO_REF_COUNT references;  
    CRYPTO_RWLOCK *lock;  
  
    void *keydata; /* Alg-specific key data */  
    EVP_SKEYMGMT *skeymgmt; /* Import, export, manage */  
}; /* EVP_SKEY */
```

EVP_SKEY API

Man EVP_SKEY

EVP_SKEY *EVP_SKEY_generate

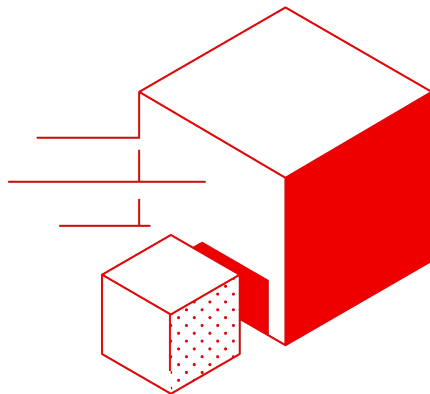
EVP_SKEY *

EVP_SKEY_import, EVP_SKEY_import_raw_key,
EVP_SKEY_import_SKEYMGMT

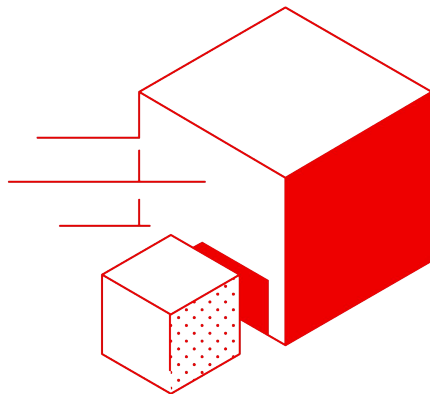
EVP_SKEY *EVP_SKEY_to_provider

int EVP_SKEY_export

int EVP_SKEY_get0_raw_key



Introducing EVP_SKEYMGMT - I



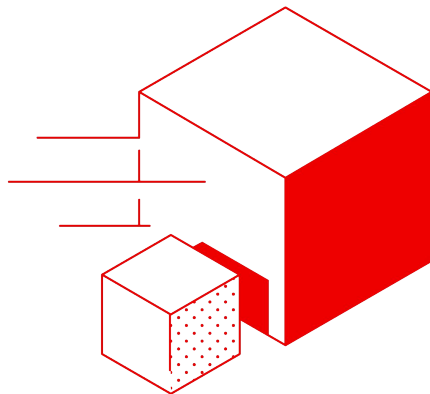
EVP_SKEYMGMT structure

```
struct evp_skeymgmt_st {  
    int name_id;  
    char *type_name;  
    const char *description;  
    OSSL_PROVIDER *prov;  
    CRYPTO_REF_COUNT refcnt;  
    ...  
    /* destructor */  
    OSSL_FUNC_skeymgmt_free_fn *free;  
} /* EVP_SKEYMGMT */;
```

Introducing EVP_KEYMGMT - II

EVP_KEYMGMT structure

```
/* Import and export routines */  
OSSL_FUNC_keymgmt_imp_settable_params_fn *imp_params;  
OSSL_FUNC_keymgmt_import_fn *import;  
OSSL_FUNC_keymgmt_export_fn *export;  
/* Key generation */  
OSSL_FUNC_keymgmt_gen_settable_params_fn *gen_params;  
OSSL_FUNC_keymgmt_generate_fn *generate;  
/* Key identifier */  
OSSL_FUNC_keymgmt_get_key_id_fn *get_key_id;
```



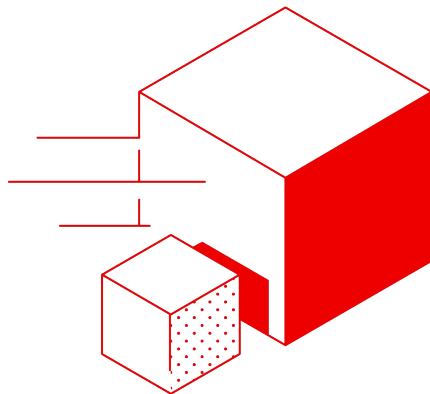
EVP_SKEYMGMT API

Man **EVP_SKEYMGMT**

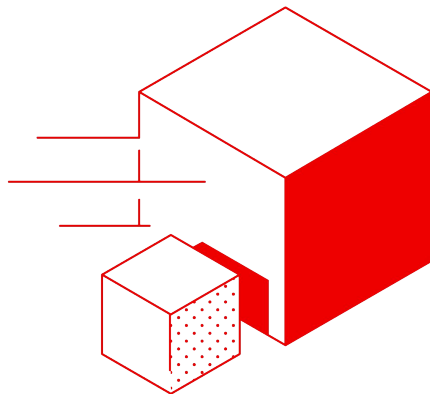
EVP_SKEYMGMT_fetch, EVP_SKEYMGMT_up_ref,
EVP_SKEYMGMT_free

EVP_SKEYMGMT_get0_provider, EVP_SKEYMGMT_is_a,
EVP_SKEYMGMT_get0_name, EVP_SKEYMGMT_get0_description

EVP_SKEYMGMT_get0_gen_settable_params,
EVP_SKEYMGMT_get0_imp_settable_params



EVP_SKEYMGMT implementation guidelines



Some sort of “inheritance”

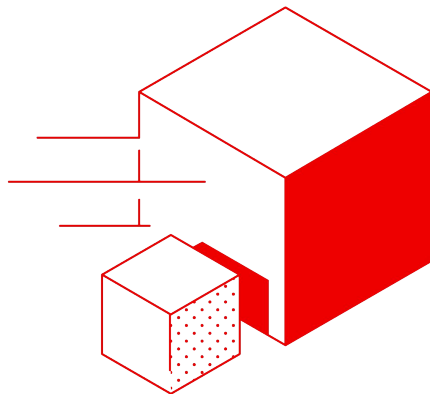
One parameterized EVP_SKEYMGMT (PKCS#11 provider)
GENERIC + AES (OpenSSL providers)
Support of raw bytes format for import

Export support

```
int EVP_SKEY_export(const EVP_SKEY *skey, int selection,  
OSSL_CALLBACK *export_cb, void *export_cbarg)
```

```
# define OSSL_SKEYMGMT_SELECT_PARAMETERS 0x01  
# define OSSL_SKEYMGMT_SELECT_SECRET_KEY 0x02
```

EVP_CIPHER an EVP_MAC API changes



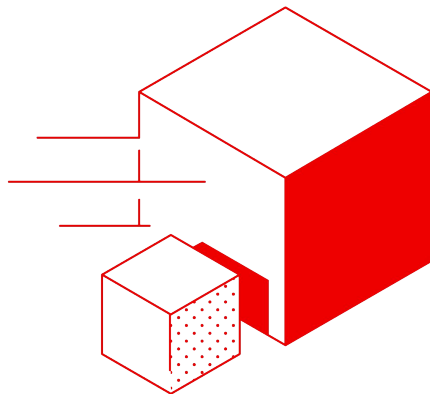
EVP_CIPHER API

```
int EVP_CipherInit_SKEY(EVP_CIPHER_CTX *ctx, const  
EVP_CIPHER *cipher, EVP_SKEY *skey, const unsigned  
char *iv, size_t iv_len, int enc, const OSSL_PARAM  
params[]);
```

EVP_MAC API

```
int EVP_MAC_init_SKEY(EVP_MAC_CTX *ctx, EVP_SKEY  
*skey, const OSSL_PARAM params[]);
```

EVP_CIPHER and EVP_MAC new callbacks



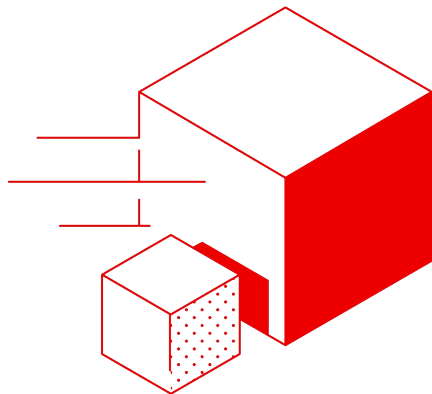
EVP_CIPHER callbacks

int **OSSL_FUNC_cipher_encrypt_key_init**(void *cctx,
void *keydata, const unsigned char *iv, size_t ivlen, const
OSSL_PARAM params[]);
... and the same for decrypt

EVP_MAC callbacks

int **OSSL_FUNC_mac_init_key**(void *mctx, const void
*key, const OSSL_PARAM params[]);

EVP_KDF an EVP_KEYEXCH API changes



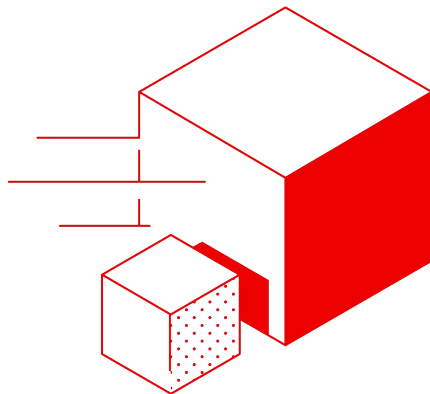
EVP_KDF API

```
int EVP_KDF_CTX_set_SKEY(EVP_KDF_CTX *ctx, EVP_SKEY
*key, const char *paramname);
EVP_SKEY *EVP_KDF_derive_SKEY(EVP_KDF_CTX *ctx,
EVP_SKEYMGMT *mgmt, const char *key_type, const char
*propquery, size_t keylen, const OSSL_PARAM params[]);
```

EVP_KEYEXCH API

```
EVP_SKEY *EVP_PKEY_derive_SKEY(EVP_PKEY_CTX *ctx,
EVP_SKEYMGMT *mgmt, const char *key_type, const char
*propquery, size_t keylen, const OSSL_PARAM params[]);
```

EVP_KDF an EVP_KEYEXCH API new callbacks



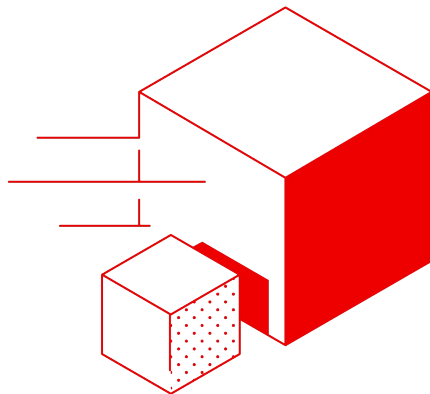
EVP_KDF API

```
int OSSL_FUNC_kdf_set_skey(void *kctx, void *skeydata,  
    const char *paramname);  
void *OSSL_FUNC_kdf_derive_skey(void *ctx,  
    const char *key_type, void *provctx,  
    OSSL_FUNC_skeymgmt_import_fn *import,  
    size_t keylen, const OSSL_PARAM params[]);
```

EVP_KEYEXCH API

```
int OSSL_FUNC_keyexch_derive_skey(void *ctx, const char  
    *key_type, void *provctx, OSSL_FUNC_skeymgmt_import_fn  
    *import, size_t keylen, const OSSL_PARAM params[]);
```

Fallback mechanisms



Old implementations, new API

Not all calls work now

Internally use `EVP_SKEY_get0_raw_key/EVP_SKEY_export`

Needs grooming

For retrieving: `EVP_SKEY_import`

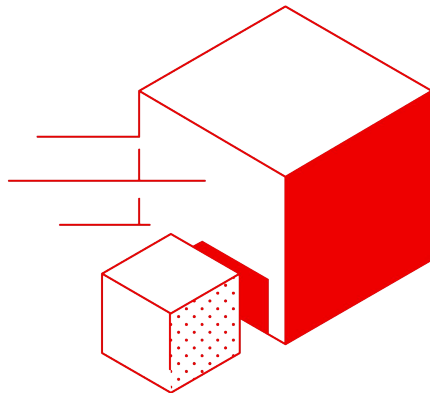
New implementations

Implement both callbacks (`do_smth/do_smth_skey`)

Providers mismatch

Export/import dance

Nearest step: STORE support



file: STORE

There is no format specification for symmetric keys

Ways forward: [RFC 6031](#)? [PKCS#12](#)?

Any sequence of bytes can be a symmetric key

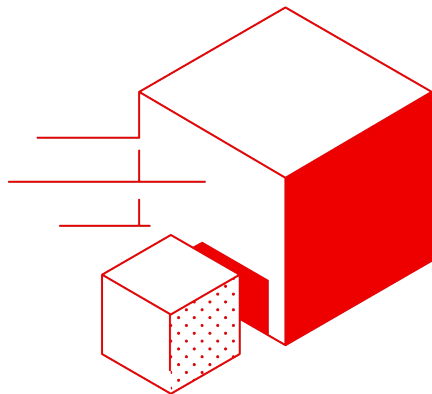
Proposal: 2048 bytes max, request EVP_SKEY explicitly

Special raw2obj decoder

pkcs11://uri

Just works out of box

Openssl command-line utilities



openssl list

-skey-managers

openssl skeyutl

-cipher, -skeymgmt, -skeyopt, -genkey

openssl enc

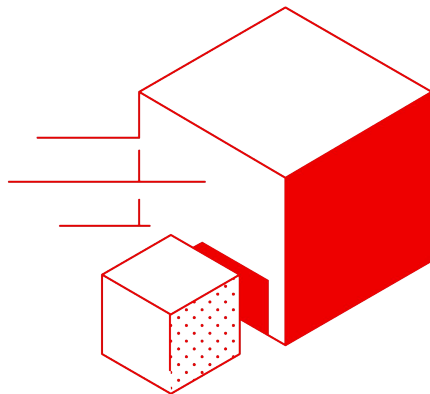
-skeymgmt, -skeyopt

After STORE changes land: -skeyuri, -storepass

openssl storeutl

After STORE changes land: -skeys

Known implementations



EVP_SKEYMGMT in OpenSSL providers

default (3.5), fips and legacy (3.6)

GENERIC-SECRET and AES

Support for test purposes: PBKDF1, ECDH KEX

PKCS#11 provider

<https://github.com/latchset/pkcs11-provider>

OpenSSL conference presentations

[Taking the OpenSSL into the PKCS#11 World and Vice Versa](#)

[Exploiting hardware-backed keys with the new EVP_SKEY API](#)

To be done

OpenSSL 4.0?

EVP_SKEY store support ([PR #28278](#))

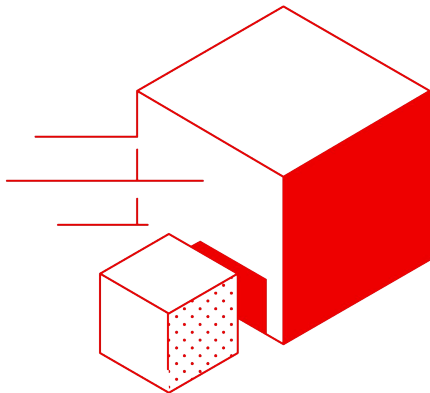
Some future

Encoders/decoders support

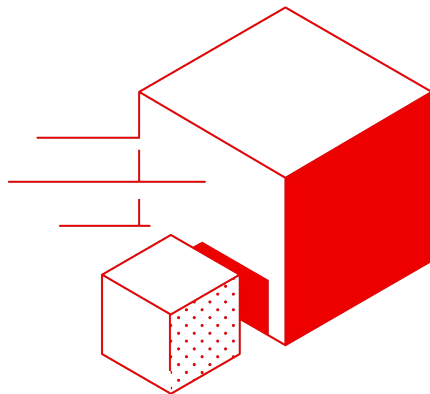
Simultaneous derivation of multiple keys (TLS)

Distant future

Use only opaque objects?



Let's cooperate



Call for SIG creation

HSM vendors

HSM users

PKCS#11 vendors

Academics community

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.



linkedin.com/company/red-hat



youtube.com/user/RedHatVideos



facebook.com/redhatinc



twitter.com/RedHat