Smartcards and DNSSEC: Simplified

IETF 96 Hackathon, Berlin, 2016

richard.lamb@icann.org

DNSSEC / HSMs

- HSM ~ $20K
- Smartcard ~15 Euro
- Most do not need speed
- KSK in HSM/card. ZSK soft and easily replaceable
- HSM and card similar security ratings (e.g., FIPS 140-2 Level 3)

OpenSC +


```
$ pkcs11-tool -M
Using slot 1 with a present token (0x1)
Supported mechanisms:
  SHA-1, digest
  SHA256, digest
  SHA384, digest
  SHA512, digest
  MD5, digest
  RIPEMD160, digest
  GOSTR3411, digest
  ECDSA, keySize={192,320}, hw, sign, other flags=0x1d00000
  ECDSA-SHA1, keySize={192,320}, hw, sign, other flags=0x1d00000
  ECDH1-COFACTOR-DERIVE, keySize={192,320}, hw, derive, other flags=0x1d00000
  ECDH1-DERIVE, keySize={192,320}, hw, derive, other flags=0x1d00000
  ECDSA-KEY-PAIR-GEN, keySize={192,320}, hw, generate_key_pair, other flags=0x1d00000
  RSA-X-509, keySize={1024,2048}, hw, decrypt, sign, verify
  RSA-PKCS, keySize={1024,2048}, hw, decrypt, sign, verify
  SHA1-RSA-PKCS, keySize={1024,2048}, sign, verify
  SHA256-RSA-PKCS, keySize={1024,2048}, sign, verify
  SHA384-RSA-PKCS, keySize={1024,2048}, sign, verify
  SHA512-RSA-PKCS, keySize={1024,2048}, sign, verify
  MD5-RSA-PKCS, keySize={1024,2048}, sign, verify
  RIPEMD160-RSA-PKCS, keySize={1024,2048}, sign, verify
  RSA-PKCS-KEY-PAIR-GEN, keySize={1024,2048}, generate_key_pair
```

Previous work

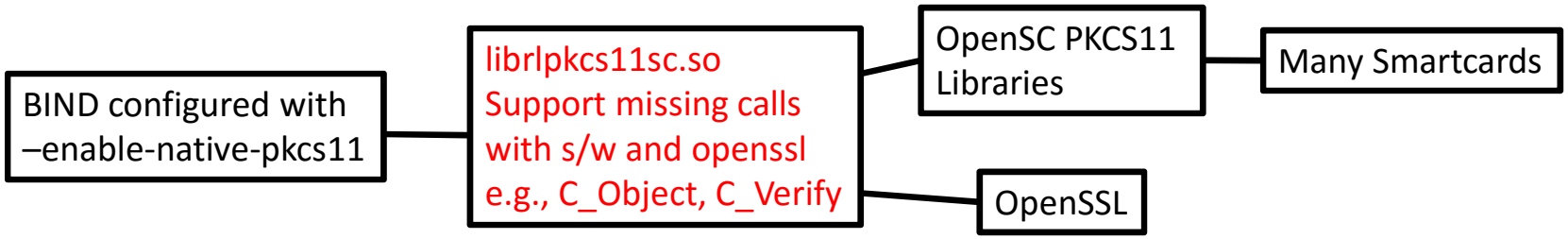
- 2006 – Jakob .SE
- 2008 – Patches to BIND (openrsa_link.c) Rick, ICANN
- 2012 – Separate support for smartcards. Rick+Luis http://ri.co.cr
- 2014 – Native PKCS11 in BIND – Hurray…sort of.
- Hackathon Goal – Open Source PKCS11 library to “glue” necessary pieces together to work with BIND with no patches.

Sample DNSSEC Key Management w/ Cards

- DNSSEC Workshop info:
  - DPS: dnssec-demo-dps.pdf
  - Key Ceremony Script: ist-dnssec-KC-demo-main.pdf
  - Log Sheets: signin-log.pdf safe-log.pdf
- Some TLDs use it.

All under: http://dnssec-deployment.icann.org/training/IST/

How It Works



Every PKCS11 call is logged: rlpkcs11sc.log

Sample steps using “it”.
Oh no, he’s going Elliptic!

```
$ sudo apt-get install opensc
$ export PKCS11_LIBRARY_PATH=/usr/lib/x86_64-linux-gnu/opensc-pkcs11.so
$ pkcs11-tool -l --keypairgen --key-type EC:prime256v1 --label ecc256key
$ pkcs11-tool -O
Using slot 1 with a present token (0x1)

Public Key Object; RSA 2048 bits
  label:      rsa2048key
  ID:         d57234301fc94fdca8a83e465ddf7e8eddd9ac4f
  Usage:      none
Public Key Object; EC  EC_POINT 256 bits
  EC_POINT:   04410431d2c27e6d305a1bc14544f7bd8b2ce587ebaf496f36b855ee15e266d
  EC_PARAMS:  06082a8648ce3d030107
  label:      ecc256key
  ID:         a7673445bb71d0f4254b822074e6148cd12f1810
  Usage:      none
```

Sample steps using “it”

```
$ ./configure --enable-native-pkcs11.
$ cc -Icryptoki -fPIC -c rlpkcs11sc.c
$ cc -shared -Wl,-soname,librlpkcs11sc.so -o librlpkcs11sc.so
rlpkcs11sc.o -lssl -lcrypto
$ echo -n "123456" > mypin    (yeah, I know ):
$ dnssec-keyfromlabel-pkcs11 -E ./librlpkcs11sc.so -l \
   "pkcs11:object=ecc256key;pin-source=mypin" -a ECDSAP256SHA256 -f
KSK hx.cds.zx.com 2>/dev/null
Khx.cds.zx.com.+013+60565
$ cat hx.cds.zx.com.0 Khx.cds.zx.com.+013+60565.key > hx.cds.zx.com
$ dnssec-signzone-pkcs11 -E ./librlpkcs11sc.so -n 1 -x -z \
    -o hx.cds.zx.com -k Khx.cds.zx.com.+013+60565 hx.cds.zx.com
Algorithm: ECDSAP256SHA256: KSKs: 1 active, 0 stand-by, 0 revoked
                            ZSKs: 0 active, 0 present, 0 revoked
hx.cds.zx.com.signed
```

Sample steps using “it”

```
$ less hx.cds.zx.com.signed
...
            120     DNSKEY  257 3 13 (
                    MdLCfm0wWhvBRUT3vYss5Yfrr0lvNrhV7hXi
                    ZtrHku74rxbdJgZU0PmjAYVeylxo65qmy+BV
                    xYVm6lLrUwoZZw==
                    ) ; KSK; alg = ECDSAP256SHA256; key id = 60565
            120     RRSIG   DNSKEY 13 4 120 (
                    20160726185027 20160626185027 60565 hx.cds.zx.com.
                    3AQVpEdFeexBs9GDWSi+ABSA2QVfM5JjosfY
                    EjJIOfKQ8KnF+7xIBiVUHeFlEpR7dzbeK9IC
                    fyZRwixFbR5dAw== )
...
```

Short KC Demo
(based on existing ceremony script)

```
hcarderase
hmakeshares
himportshare dkek-share-X.pbe
export DOMAIN=ietf  (export TEST=yes)
hgenkskec
hcardshow
hwrapkey  (hcarderase, himportshare, hunwrapkey)
hgenzskec  (hcardrng if needed)
hcardsignec
(remove card)
dnssec-dsfromkey *.key
hsignzoneec
dig +dnssec +multi -t DNSKEY ietf @127.0.0.1
cat tmp/namedb/signemd.out
```

Thanks, Q+A
updates at: http://ri.co.cr

Links:

https://github.com/OpenSC/OpenSC/wiki/SmartCardHSM

Maybe a github for this?

Every call logged?