

EOSC-hub RCauth

- *IGTF accredited*
- *Currently hosted/run by NIKHEF*
- *Create cloned CAs at STFC and GRNET*
 - *While retaining IGTF accreditation*
 - *Kind of like ESNET's distributed whatsamabobs*
- *Single service, single governance (PMA)*
- *Use for high availability-resilience*
- *Maybe additional usefulness w.r.t. private key mgmt*
 - *I.e. can learn something about archiving keys*

1/21/2019

Jens Jensen

2

EOSC-hub Task Overview

- *5.1.7 RCauth Governance and operations*
 - *5.1.7.1 Documentation*
 - *5.1.7.2 Governance, PMA*
 - *5.1.7.3 Operations – support, monitoring and performance, compliance*
- *5.1.8 Resilient RCauth*
 - *5.1.8.1 Establish RCauth key at all sites*
 - *5.1.8.2 Refactoring of RCauth code (if necessary)*
 - *5.1.8.3 Central services high availability*
 - *5.1.8.4 Update governance infrastructure (e.g. support)*
 - *5.1.8.5 Pathfinder*

1/21/2019

3

EOSC-hub Copying the key

- Process agreed in outline at PMAs May '18 and Sep '18
- Say k is the key
 - Generate random integers of same size as k , a , b
 - Key is split in three independent parts, a , b , $a \oplus b \oplus k$
 - Where " \oplus " denotes bitwise XOR
 - No one or two parts alone have any information about the key (in an information theoretical sense)
 - Recipient \otimes s all the pieces together to recover k

1/21/2019

4

EOSC-hub Randomness

- Randomness
 - jensen@ganesh[2]1% openssl rand 12 >file
 - jensen@ganesh[2]2% ls -l file
 - -rw-r--r-- 1 jensen jensen 12 Jan 18 11:22 file
 - jensen@ganesh[2]3% od -x file
 - 0000000 dda6 ba48 93ea 9007 30f3 c9fe
 - 0000014
- Needs strong cryptographic randomness
 - openssl rand -engine xyzy 128
- Assuming FIPS140-2 includes checks for randomness
- Does randomness use the engine, or is the engine just used to seed, or is the engine not used at all?
- Need to consult the Documentation™

1/21/2019

5

EOSC-hub Documentation (from 1.0.1t)

```
while (num > 0) {
    unsigned char buf[4096];
    int chunk;

    chunk = num;
    if (chunk > (int)sizeof(buf))
        chunk = sizeof buf;
    r = RAND_bytes(buf, chunk);
    if (r <= 0)
        goto err;
    if (!hex)
        BIO_write(out, buf, chunk);
    else {
        for (i = 0; i < chunk; i++)

int RAND_bytes(unsigned char *buf, int
num)
{
    const RAND_METHOD *meth =
RAND_get_rand_method();
    if (meth && meth->bytes)
        return meth->bytes(buf, num);
    return (-1);
}
```

```
BIO_write(out, buf, chunk);
else {
    for (i = 0; i < chunk; i++)
        BIO_printf(out, "%02x", buf[i]);
}
num -= chunk;
}
```

1/21/2019

6

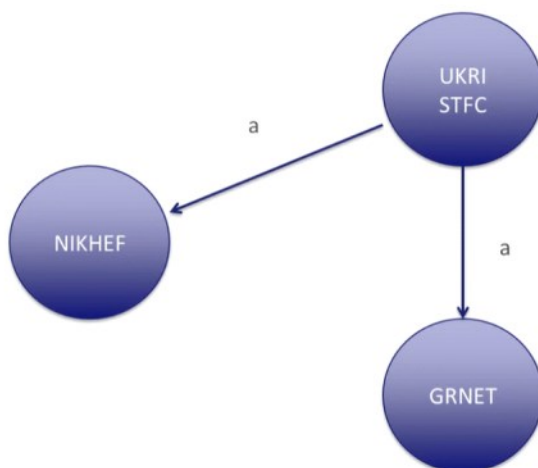
EOSC-hub Down the Rabbit Hole...

```
const RAND_METHOD *RAND_get_rand_method(void)
{
    if (!default_RAND_meth) {
#ifdef OPENSSL_NO_ENGINE
        ENGINE *e = ENGINE_get_default RAND();
        if (e) {
            default_RAND_meth = ENGINE_get RAND(e);
            if (!default_RAND_meth) {
                ENGINE_finish(e);
                e = NULL;
            }
        }
    }
    if (e)
        funct_ref = e;
    else
#endif
        default_RAND_meth = RAND_SSLeay();
}
return default_RAND_meth;
```

1/21/2019

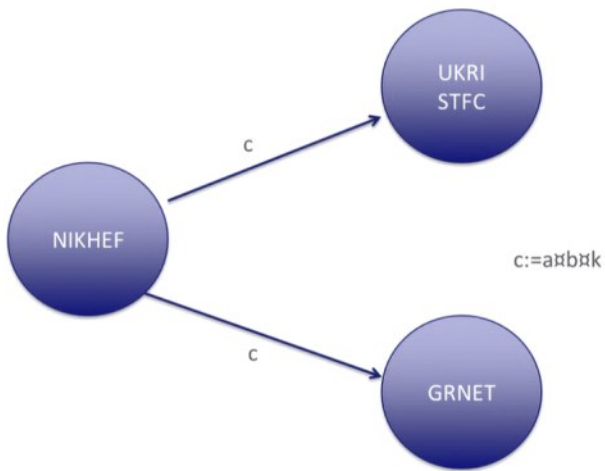
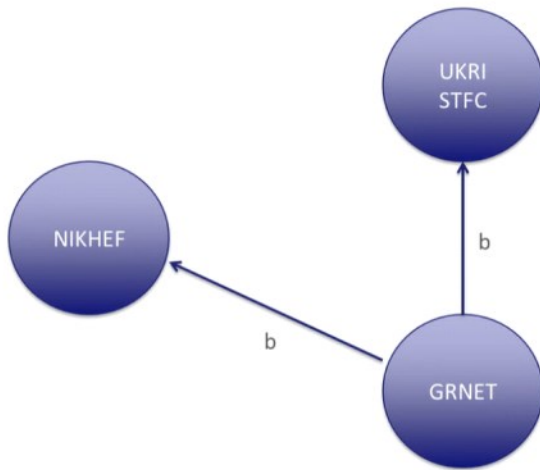
7

EOSC-hub Chinese Whispers



1/21/2019

9



- Each of a , b , c should be exchanged using different means
 - Physical in person delivery
 - Sent by encrypted email
 - Sent by snail mail
- If a piece is lost/compromised, regenerate it and try again
 - Or accept the compromise: no two pieces have any information about the key

- If a piece $\{a,b,c\}$ is exchanged on paper:
 - Use QR code
 - OCR it
 - Or type it in by hand if it's short
 - Not hand written...
- Ideally use offline methods (no office printer spooler)
- A 2048 bit key is about 1200 bytes in DER encoding
 - Nearly 1700 characters in PEM
- However,...

SEQUENCE			
INTEGER	version		00
INTEGER	modulus (PUBLIC)		$p \cdot q$
INTEGER	exponent (PUBLIC)		e
INTEGER	private exponent		$e^{-1} \pmod{\text{lcm}(p-1, q-1)}$
INTEGER	prime1		p
INTEGER	prime2		q
INTEGER	exp1		$e^{-1} \pmod{p-1}$
INTEGER	exp2		$e^{-1} \pmod{q-1}$
INTEGER	inv		$q^{-1} \pmod{p}$

This is a level 1 only
sequence of integers

1/21/2019

14

EOSC-hub Corollary

- Can reconstruct the private key from
 - The public key
 - And one of the primes
- See CAOPS-WG, OGF 23, Barcelona (June, 2008)
- If the modulus is 2048 bits, each prime is 1024 bits
 - Very likely
 - Definitely one prime is ≤ 1024 bits !
- \Rightarrow Protect the prime!

1/21/2019

15

- Need to understand the ASN.1
 - Which is complicated?
 - Except here, it is not at all... [Larmouth (2000)]
 - TLV (Tag, Length, Value) – all at a single level `SEQUENCE SIZE (9) OF INTEGER`
- TAG 0x30 SEQUENCE (class 0, constructed, number 16)
- 0x82 + 2 bytes of total length of value
- Each value is itself TLV encoding of `INTEGER`
 - TAG 0x02 (class 0, primitive, number 2)
 - ...

1/21/2019

16

- Need to understand the ASN.1
 - Which is complicated?
 - Except here, it is not at all... [Larmouth (2000)]
- TLV (Tag, Length, Value) – all at a single level `SEQUENCE SIZE (9) OF INTEGER`
- We need only two Tags
 - TAG 0x30 SEQUENCE (class 0, constructed, number 16)
 - TAG 0x02 (class 0, primitive, number 2)



1/21/2019

17

- Length
 - One byte of Length for ≤ 127
 - 0x81 plus one byte of length for $128 \leq \text{length} < 256$
 - 0x82 plus two bytes of Length for ≥ 256
- INTEGER Value has leading 0x00 if MSB is $\geq 0x80$
 - (For positive integers)
 - “top nine bits must not be the same”
- From BER to DER
 - All Ls are explicit, using the shortest encoding
 - All Vs use the shortest permitted encoding

- *Repeat the exercise with Pathfinder?*
 - *DOGWOOD => RCauth*
 - *BIRCH => Pathfinder*
- *Would rekey Pathfinder upon accreditation*
- *Could do shared key generation*
 - *A la Diffie Helman except still RSA*
 - *No need to rerun distribution process*
- *Alternatively use RCauth key to share Pathfinder*
 - *Should be OK...*

- Interesting enough exercise to share
 - Crypto random
 - Key splitting
 - (done right, albeit with some technical requirements)
- CAOPS?

Jens Jensen UKRI STFC



EOSC-hub

 eosc-hub.eu  [@EOSC_eu](https://twitter.com/EOSC_eu)