SDU
University of
Southern Denmark

# Cryptographic Engineering

## Introduction to Cryptography (Part II)

---

Ruben Niederhagen
(based on content by Norman Lahr and Richard Petri)

Department of Mathematics and Computer Science (IMADA)

# Symmetric Schemes (continued)

## Message Authentication Codes (MACs)

# What is a Message Authentication Code?



- A Message Authentication Code is function

$$MAC_k : \{0,1\}^* \to \{0,1\}^n$$

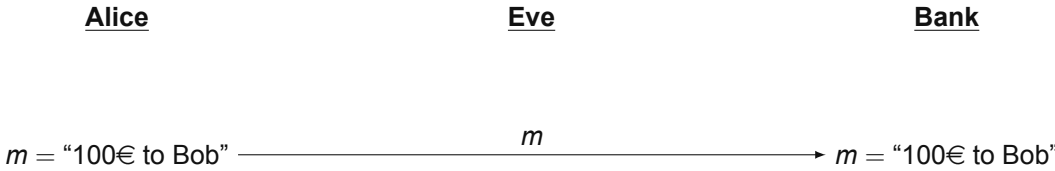
  that generates a cryptographically secure authentication tag $t$ for a given message $m$.
- The signing and verifying parties share a symmetric key $k$:

$$t = MAC_k(m).$$

- A MAC provides message authenticity and integrity.
- It does *not* provide non-repudiation since both parties share the same MAC key.

Department of Mathematics
and Computer Science (IMADA)

**Financial transaction:**



**Financial transaction:**

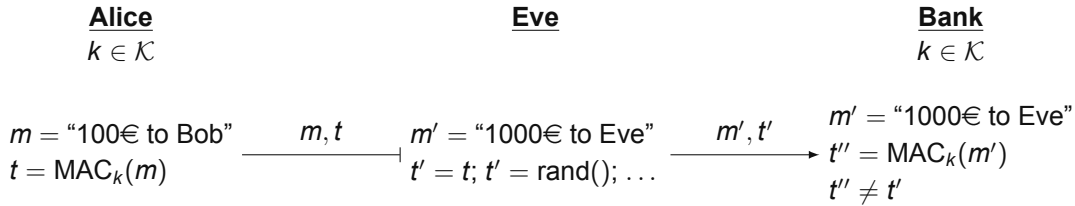
**Alice** **Eve** **Bank**

$$m = \text{“100€ to Bob”} \xrightarrow{\quad m \quad} m' = \text{“1000€ to Eve”} \xrightarrow{\quad m' \quad} m' = \text{“1000€ to Eve”}$$

**Financial transaction:**

| **Alice** | **Eve** | **Bank** |
|---|---|---|
| $k \in \mathcal{K}$ | | $k \in \mathcal{K}$ |
| $m =$ "100€ to Bob" | $m, t$ | $m =$ "100€ to Bob" |
| $t = \mathrm{MAC}_k(m)$ | $\longrightarrow$ | $t' = \mathrm{MAC}_k(m)$ |
| | | $t = t'$ |

**Financial transaction:**



# MACs from Hash Functions: HMACs



- Cryptographic hash functions and their one-wayness property can be used to build a MAC.
- Idea: A secret key $k$ is hashed together with the message $m$.
- Intuitive constructions are the prefix (potentially insecure) or suffix ("weak") concatenation.

**Secret prefix MAC (potentially insecure):**

$$t = MAC_k(m) = h(k||m)$$

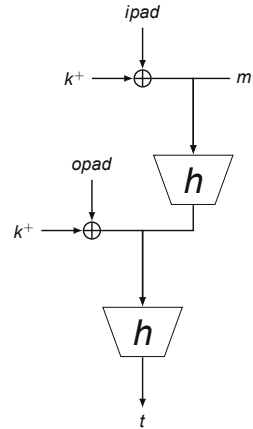
**Secret suffix MAC ("weak"):**

$$t = MAC_k(m) = h(m||k)$$

Department of Mathematics
and Computer Science (IMADA)

- Developed by Bellare, Canetti und Krawczyk in 1996.
- Idea: “Wrapper” around hash function $h$.
- The key $k$ is not concatenated but integrated into the message $m$.
- Two hash function instances to increase the collision resistance: inner and outer.
- Common key $k^+$ for both hash functions; expanded by 0s, aligned to the hash function block size.

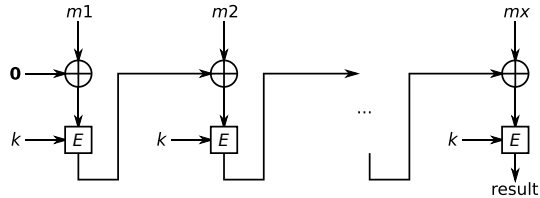
**HMAC construction:**




$$HMAC_k(m) = h((k^+ \oplus opad)||h((k^+ \oplus ipad)||m))$$

- A CBC-MAC is based on the Cipher Block Chaining (CBC) block mode.
- It requires a block cipher, e.g., AES.
- The $IV$ is random, can be public.
- The message is split into aligned blocks: $m = m_0, m_1, \ldots$.
- The authentication tag is the last ciphertext in the chain.

**CBC-MAC construction:**



Department of Mathematics
and Computer Science (IMADA)

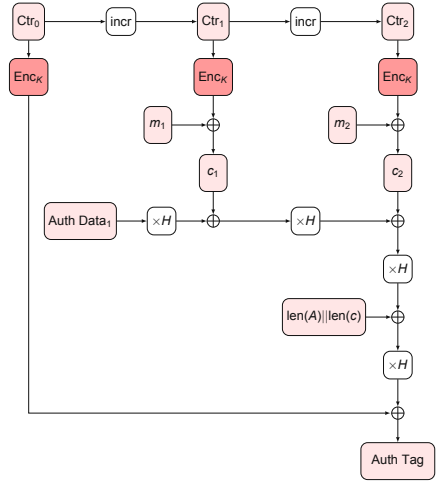
- Goal: Authentic and confidential data.
- MAC-then-Encrypt:
  - Authenticate the message; encrypt the message and the authentication tag.
  - Send ciphertext.

  $\Rightarrow$ Padding oracle attack possible!
- Encrypt-then-MAC:
  - Encrypt the message; authenticate the ciphertext.
  - Send ciphertext and authentication tag.
- Encrypt and MAC:
  - Authenticate and encrypt the message.
  - Send ciphertext and authentication tag.

Department of Mathematics
and Computer Science (IMADA)

- Couner Mode + Authentication
  $\rightarrow$ Authenticated Encryption (AE).
- Constant $H = ENC_k(0)$.
- Multiplication computed in a Galois field.
- GMAC: MAC without encryption output.

**GCM construction:**



# Asymmetric Schemes

## Encryption

Department of Mathematics
and Computer Science (IMADA)

**Definition (Group)**

A *group* consists of a set $\mathcal{G}$ and a binary operation $\cdot$ with the following properties:

- For all $a, b \in \mathcal{G}$ it holds that also $a \cdot b \in \mathcal{G}$, the group is *closed* under the operation $\cdot$.
- For all $a, b, c \in \mathcal{G}$ it holds that $(a \cdot b) \cdot c = a \cdot (b \cdot c)$, i.e. the *associative property*.
- There is an *identity* (or *neutral*) element $1 \in \mathcal{G}$ such that for any $a \in \mathcal{G}$ it holds that $1 \cdot a = a \cdot 1 = a$.
- For each $a \in \mathcal{G}$ there is an *inverse* $a^{-1}$ such that $a \cdot a^{-1} = a^{-1} \cdot a = 1$.
- Optional: If for all $a, b \in \mathcal{G}$ it holds that $a \cdot b = b \cdot a$, then the group is called *commutative* (or *abelian*).

**Additive and multiplicative groups:**

- The group operation can also be written as $+$, then its usually called an *additive group* instead of a *multiplicative group*.
- The inverse in an additive group is usually denoted as $-a$ instead of $a^{-1}$.

**Definition (Group Order)**

The *group order* $\#\mathcal{G}$ is the number of elements in the group.

**Definition (Group Generator)**

A *generator* $g \in \mathcal{G}$ generates all elements of the group via exponentiation, i.e. $\mathcal{G} = \{1,\ g,\ g \cdot g,\ g \cdot g \cdot g,\ \ldots\}$.

Department of Mathematics
and Computer Science (IMADA)

**Definition (Field)**

A *field* is a set $\mathbb{F}$ and two binary operations $+$ and $\cdot$ with the following properties:

- $(\mathbb{F}, +)$ are an abelian group with neutral element 0.
- $(\mathbb{F}\backslash\{0\}, \cdot)$ are an abelian group with neutral element 1 and $1 \neq 0$.
- For all $a, b, c \in \mathbb{F}$ it holds that $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$ and $(b + c) \cdot a = (b \cdot a) + (c \cdot a)$, i.e. the *distributive property*.

**Common fields in cryptography:**

- The *finite field* $\mathbb{F}_p$ or GF($p$) ("Galois Field"), i.e. *modular arithmetic* with integers modulo some prime $p$, is a very common field in cryptography.
- Another common field is $\mathbb{F}_{2^n}$ or GF($2^n$), i.e. *polynomial arithmetic* with polynomials modulo some irreducible polynomial $R$.

# What is Asymmetric Encryption?



**Definition:**

- An encryption scheme with different keys and the en- and decryption functions $E_e$ and $D_d$ is said to be an *asymmetric* or *public-key encryption scheme*.
- Such a scheme is *secure* if it is infeasible to compute the decryption key *d* from the encryption key *e*.
- The encryption key *e* is also called the *public key*, the decryption key *d* the *private key*.

**Notes:**

- For symmetric encryption the key *k* is usually called the *secret key*.
- The public key can be publicly known and allows anyone to encrypt messages that only the owner of the private key can decrypt.
- Asymmetric schemes are usually based on some mathematical problem or operation which is easy to calculate in one direction, but hard to reverse.

Department of Mathematics
and Computer Science (IMADA)

**Definition (Factorization)**
There are no known classical algorithms for finding the prime factorization of integers in polynomial time.

- There are, however, sub-exponential and quantum algorithms!

**Definition (Discrete Logarithm)**
In a group $(G, \cdot)$ with generator $g \in G$, find $a$ for a given $x \in G$ such that
$g^a = \underbrace{g \cdot g \cdots g}_{\text{a times}} = x$.

- Also defined for additive groups: Find $i$ such that $[i]g = \underbrace{g + g + \cdots + g}_{\text{i times}} = x$.

The Discrete Logarithm Problem (DLP) is related to the Factorization Problem and typically about as hard.

Other hard problems: Shortest vector problem, decoding problem, …

Department of Mathematics
and Computer Science (IMADA)

Naivly computing exponentiation $b = g^a$ of $g$ by $a$ using $a$ multiplications can be expensive.

A better approach:

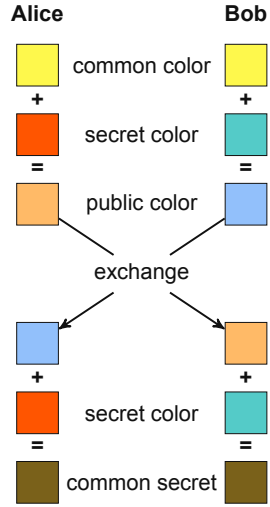
**Require:** $a$ in binary form: $a = (a_{n-1}, \ldots a_1, a_0)_2$ with $a_i \in \{0, 1\}$
**Ensure:** $b = g^a$

```
b ← 1
for i ∈ (n − 1, . . . , 1, 0) do
    b ← b^2                    // square
    if a_i = 1 then
        b ← b · g              // multiply
    end if
end for
```

# Diffie-Hellman Key Exchange



- The first known public-key cryptosystem.
- Allows two parties to establish a common secret over a public channel.
- Works with any algebraic group $(\mathcal{G}, \cdot)$ with generator $G \in \mathcal{G}$ as follows:
  - Each party picks a random integer $a, b > 1$ ($a$ for Alice, $b$ for Bob).
  - Each party computes their public key $A = G^a$ (or $B = G^b$)
  - $A$ and $B$ are exchanged.
  - Each party computes the shared secret $K = B^a$ (or $K = A^b$).



**Key generation:**

- Pick two random and distinct primes $p$ and $q$.
- Compute $n = pq$ and $\phi(n) = (p-1)(q-1)$.
- Select integer $e$ with $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$
- Compute the multiplicative inverse $d$ of $e$ such that $ed \equiv 1 \mod \phi(n)$.
- The public key is then $(n, e)$ and the private key $d$.

**En- and decryption:**

- Encode a message as integer $m \in [0, n)$ and compute ciphertext $c = m^e \mod n$.

$\rightarrow$ There are a lot of pitfalls when encoding the message, usually a special padding algorithm is used.

- To decrypt the ciphertext compute the message $m = c^d \mod n$.

Department of Mathematics
and Computer Science (IMADA)

- Public-key schemes are usually an order of magnitude slower (e.g., $\approx$ 10000 vs $\approx 10M$ cycles)!
- Public-Key Cryptography is usually not applied to messages directly.
- Instead a *hybrid* approach with symmetric schemes is used:
  Public-key crypto is used to encrypt a secret key for a symmetric block cipher.

# Asymmetric Schemes

## Signature Schemes

# Different Types of Authentication



**Identification:**
A method for determining and verifying the identity of a communication partner.

Identity check during an ongoing process, for example, during login.

**Data origin authentication:**
A method for determining and verifying the origin of information.

Binding an identity to a Message, for example, with a *signature*.

Department of Mathematics
and Computer Science (IMADA)

**Digital signature scheme:**

- A signature function $S_A(m) = s$ produces a *signature* $s$ for a message $m$ depending on some identity $A$.
- A verification function $V_A(m, s)$ verifies the validity of a message-signature pair.

**Properties:**

- Only the owner of the identity should be able to produce signatures, but anyone should be able to verify them.
- A signature should not be reputable (*non-repudiation*).
- A signature should not be malleable (possible with some weak RSA paddings).

- RSA can also easily be used as a signature scheme.
- The application order of the keys are simply reversed: $d$ is now private, $e$ is public.
- The message $m$ is “encrypted” with the private key and can be “decrypted” with the public key:

$$s = m^d \bmod n \quad \longleftrightarrow \quad m = s^e \bmod n$$

- Hybrid approach is used as well:
  The hash of a message is signed using a signature scheme.

$\Rightarrow$ Proper padding is extremely important, see RFC3447.

Department of Mathematics
and Computer Science (IMADA)

**Keygen:**

- Select a random integer $a \in [1, \#\mathcal{G})$, which is the private key.
- Compute $y = g^a$ which is the public key.

**Signing:**

- Pick random nonce $k \in [1, \#\mathcal{G})$.
- Compute: $r = g^k \bmod \#\mathcal{G}$
- Compute: $s = k^{-1} \cdot (H(m) + a \cdot r) \bmod \#\mathcal{G}$
- Ouput $(r, s)$ as the signature for message $m$.

**Verifying:**

- Verify $0 < r < n$ and $0 < s < n$.
- Compute: $u_1 = H(m)s^{-1} \bmod \#\mathcal{G}$
- Compute: $u_2 = rs^{-1} \bmod \#\mathcal{G}$
- Compute: $r' = g^{u_1} \cdot y^{u_2} \bmod \#\mathcal{G}$ $= g^{H(m)s^{-1} + ars^{-1}}$
- Verify $r = r'$, reject otherwise.

# Digital Signature Algorithm (DSA) Notes



- The concept of DSA can be used with any group with hard discrete logarithm.
- Note that you have to make calculations with group elements as well as integers modulo the group order.
- The nonce $k$ has to be properly (uniformly) random.
- Alternatively, one can generate $k$ with a specialized HMAC construction, see RFC6979.

# What happens if $k$ is not a nonce:



ars TECHNICA SUBSCRIBE SIGN IN

GAMING & CULTURE —

## PS3 hacked through poor cryptography implementation

A group of hackers named fail0verflow revealed in a presentation how they ...

CASEY JOHNSTON - 12/30/2010, 6:25 PM

A group of hackers called fail0verflow claim they've figured out a way to get better control over a PlayStation 3 than ever before. After they worked through a number of Sony's security measures, they found the keystone to gaining access to the system's innards was the PS3's poor use of public key cryptography.

At the Chaos Communication Conference 27C3, the team gave a 45-minute presentation on the methods they used to work through the PS3's various security levels, which include a chain of trust, a hypervisor, and signed

[https://arstechnica.com/gaming/2010/12/ps3-hacked-through-poor-implementation-of-cryptography/]

# Biased Nonce Sense: Lattice Attacks against Weak ECDSA Signatures in Cryptocurrencies

Joachim Breitner[1][0000−0003−3753−6821] and Nadia Heninger[2]

[1] DFINITY Foundation, Zug, joachim@dfinity.org
[2] University of California, San Diego, nadiah@cs.ucsd.edu

**Abstract.** In this paper, we compute hundreds of Bitcoin private keys and dozens of Ethereum, Ripple, SSH, and HTTPS private keys by carrying out cryptanalytic attacks against digital signatures contained in public blockchains and Internet-wide scans. The ECDSA signature algorithm requires the generation of a per-message secret nonce. If this nonce is not generated uniformly at random, an attacker can potentially exploit this bias to compute the long-term signing key. We use a lattice-based algorithm for solving the hidden number problem to efficiently compute private ECDSA keys that were used with biased signature nonces due to multiple apparent implementation vulnerabilities.

**Keywords:** Hidden number problem, ECDSA, Lattices, Bitcoin, Crypto

# Extract the private key from two signatures



- If $k$ is the same for two signatures $(r_1, s_1)$ and $(r_2, s_2)$, we can compute $k$.

$$\begin{aligned}\frac{s_1 - s_2}{z_1 - z_2} &= \frac{k^{-1} \cdot (z_1 + a \cdot r_1) - k^{-1} \cdot (z_2 + a \cdot r_2)}{z_1 - z_2} \\ &= \frac{k^{-1} \cdot (z_1 - z_2)}{z_1 - z_2} = k^{-1}\end{aligned}$$ (We know $z_1$ and $z_2$ since we now the messages)

- If we can compute $k$, we can compute the private key $a$ from a signature.

$$\begin{aligned}\frac{s_1 \cdot k - z_1}{r_1} &= \frac{k^{-1} \cdot (z_1 + a \cdot r_1) \cdot k - z_1}{r_1} \\ &= \frac{z_1 + a \cdot r_1 - z_1}{r_1} = a\end{aligned}$$

- If $k$ is not the same, but not properly random, we need more signatures and use a "lattice attack" as in the Breitner/Heninger attack.

# Random Numbers

Department of Mathematics
and Computer Science (IMADA)

Random numbers are (implicitly) used in nearly every cryptographic application.

**Block and stream ciphers**

- An encryption function takes a binary key $k \in \{0,1\}^n$ of length $n$ … …

**RSA encryption scheme**

- Pick two random and distinct primes $p$ and $q$.

  …

- Secret, private, and public keys,
- initialization vectors,
- nonces,
- salts,
- seeds,
- for masking, blinding, and hiding implementations (later),
- …

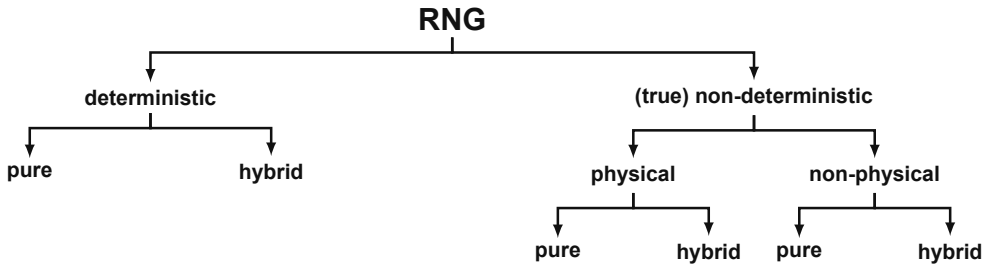
# Requirements of cryptographic Randomness

- The generation of random numbers is also referred to as “experiment”.
- *Ideal random* experiments are unpredictable, independent and unbiased (*ideal randomness*).
- Unpredictability: The observable outcome of the experiment is (to a certain extent) unknown before it is conducted. After the experiment has been performed, the degree of uncertainty depends on the ability to observe the outcome.
  - We denote the outcome of an experiment as random if it is unpredictable.
  - Entropy quantifies the amount of unpredictability relative to the observer.
- Independence: The outcomes of previous experiments do not influence the outcome of the current experiment.
- No bias: Each admissible outcome has the same chance of occurring.

$\Rightarrow$ A *good* entropy source is the core of each random number generator (RNG).

# True vs. Deterministic Random Number Generators

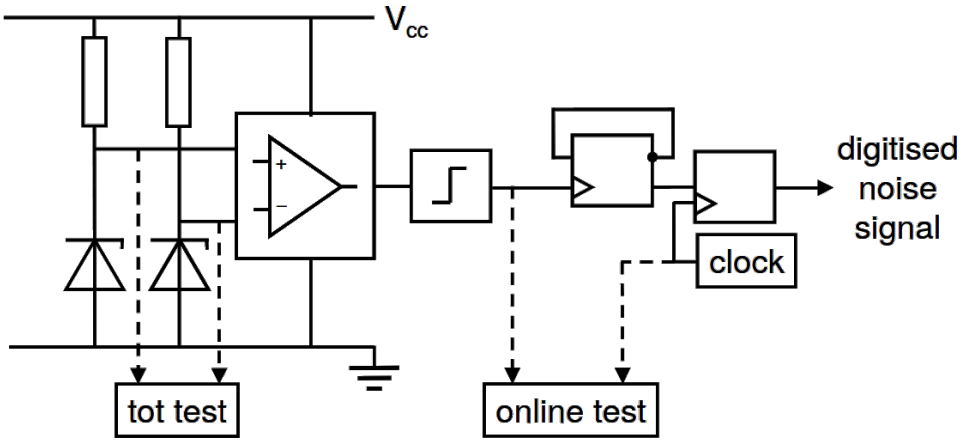
- An RNG consists of a non-deterministic part (*entropy source*) that generates non-predictable digital data, and a deterministic part that generates the output sequence of the RNG (*random numbers*) from this data.
- The non-deterministic part of the RNG exploits a physical entropy source or any other kind of non-physical entropy source to generate a raw random number sequence, which is deterministically post-processed.



- The core of any Physical True Random Number Generator (PTRNG) is the entropy source.
  - By exploiting an analogue signal, a digitization mechanism generates a sequence of digital “raw” data (raw random numbers).
  - A post-processing algorithm transforms the raw data to internal random numbers.
- Time-discrete physical entropy source are, e.g.,
  - Radioactive atomic disintegration: The number of detected particles per time interval follows a Poisson distribution.
- Analogue physical entropy sources are, e.g.,
  - Thermal resistive entropy: The voltage between resistors varies randomly due to vibration of atoms.
  - Diode breakdown entropy: The reverse current through diodes varies randomly due to tunnelling of electrons.
  - Sampling a fast free running oscillator by a lower frequency oscillator generates a random bit signal (jitter).

Department of Mathematics
and Computer Science (IMADA)

**RNG design with two noise diodes [Killmann et al., 2008]:**



- A Non-physical True Random Number Generator (NPTRNG) uses external signals as entropy source.
- The concept of randomness is the lack of information about processes and their outcomes.
- External entropy sources are, e.g.,
  - Processes as disk I/O operations and interrupts (e.g., Linux RNG /dev/random).
  - System data as tick counter since system boot, process and thread IDs, and current local time (e.g. in MS Windows CE Enhanced Cryptographic Provider).
  - Human interaction as mouse movement and key strokes (e.g. PGP key generation).
- A huge amount of data from different source and post-processing (e.g. by a hash function) are required.

## /dev/random

```
> cat /dev/random | xxd
00000000: c24f 0638 8380 18e0 588f b93e 4fa7 1788  .O.8....X..>O...
00000010: c95f 5e8e f954 158c b28a 6b15 75e4 ef98  ._^..T....k.u...
00000020: 87ec 46b1 77f0 8a5d e910 7e0c 4582 b7d5  ..F.w..]..~.E...
00000030: 578e 1526 d790 ce7d 7680 5c4f b226 5407  W..&...}v.\O.&T.
00000040: 7dfd aa76 41c7 0c20 c299 add0 0701 7194  }..vA.. ......q.
00000050: 2640 07a9 deb4 7aa8 590f 21a5 c09c fc13  &@....z.Y.!.....
00000060: c1de bd68 2ed2 78f0 27a2 8fe0 2e7e f4e2  ...h..x.'....~..
00000070: 4d4e b8a7 595c 64ab 3c8e 9656 5fb7 a004  MN..Y\d.<..V_...
00000080: 32b0 4597 b1a5 144d 1da6 f85c 9017 a894  2.E....M...\....
00000090: 92fb a584 dbd4 5aff 49b6 c445 b147 c984  ......Z.I..E.G..
000000a0: 877d da6f f25b b8c5 6c97 3d9e 2170 ac46  .}.o.[..l.=.!p.F
000000b0: d6a3 8c71 9bfd 2f4f 59cd 7ddf b5d4 2a67  ...q../OY.}...*g
000000c0: 6e40 2322 09f1 d0de 6d38 5db5 5756 bd61  n@#"....m8].WV.a
000000d0: 0fb2 3bb7 cadb cac9 087d 4dcf d0d1 f472  ..;......}M....r
000000e0: 59b6 d77c 8693 0269 acd7 a101 fc01 a6ea  Y..|...i........
000000f0: bc10 7987 d45f d533 8955 110f f08a a0ac  ..y.._.3.U......
```

# Example of Entropy Pool



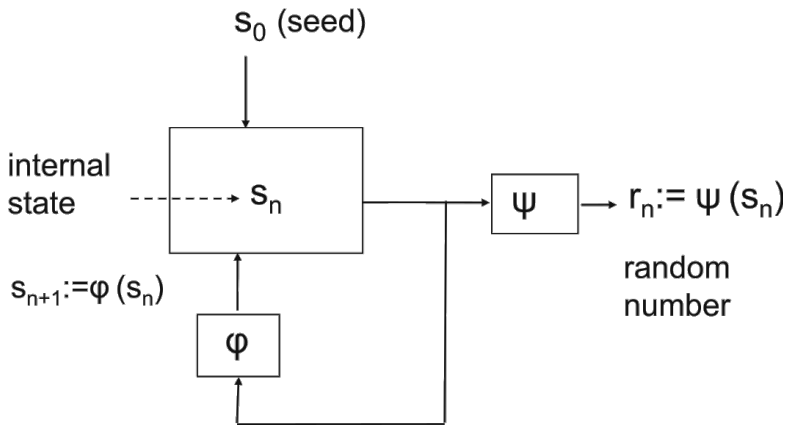
## Kernel Entropy Pool

```
> cat /proc/sys/kernel/random/entropy_avail
3771
```

- A Deterministic or Pseudo Random Number Generator (DRNG, PRNG) generates pseudo random sequence with a deterministic algorithm and starts with a seed.
- Examples are the usage of hash functions or block ciphers.
- The output depends on the internal state and possibly on additional input.
- Gain of entropy from the seed and possibly from additional input during the operational work (reseeding or refreshing).

Department of Mathematics
and Computer Science (IMADA)

**Pure DRNG [W.Schndler in Cryptographic Engineering, 2009]**



- A Hybrid Random Number Generator combines the design principles of true and deterministic RNGs.
- It consists of an entropy source and a deterministic part.
- The entropy source must generate at least so much entropy that a perfect post-processing algorithm might generate an ideal output sequence.

# Example of an Hybrid RNG



## /dev/urandom

```
> cat /dev/urandom | xxd
00122ab0: 76fb c2be cf9b 5cea 85b7 0745 19f2 248c  v.....\....E..$.
00122ac0: 032e 2b5f d344 5381 471b a4ba 200c ba8d  ..+_.DS.G... ...
00122ad0: 4142 a687 f312 c305 699b 1e4d d277 175f  AB......i..M.w._
00122ae0: 6898 654e 211a 630c b850 b345 e0aa 7d35  h.eN!.c..P.E..}5
00122af0: eccb 2bed 4ea0 5529 aea0 a02a c853 2285  ..+.N.U)...*.S".
00122b00: 6a4f 523e bb8b de0c 6cc5 9dc5 6d37 3c5d  jOR>....l...m7<]
00122b10: 9e8d 1681 f70f b88f 4b2f c2ab f1a2 35a7  ........K/....5.
00122b20: 3a24 bbcd 86f8 46ce 1a35 d230 6791 e876  :$....F..5.0g..v
00122b30: 779b 87dc 0f5f b18a 6bc6 4081 451d 1b0c  w...._..k.@.E...
00122b40: d216 e88f cff9 b9b7 02fb 41a1 a600 383d  ..........A...8=
00122b50: b287 e660 080c 5a42 4686 c55f 41db 9388  ...`..ZBF.._A...
00122b60: 7fd2 925d 41d3 bd85 433a dcd0 f7c7 d46b  ...]A...C:.....k
00122b70: 99f5 a039 9e72 bd6c f6c5 3906 bc05 d443  ...9.r.l..9....C
00122b80: b479 6fb4 6f80 61fd ec07 bc21 ddd7 85b3  .yo.o.a....!....
00122b90: 669a f2e8 224e 2769 b663 8e53 76c5 808f  f..."N'i.c.Sv...
00122ba0: f764 f885 79e0 b0cc 69eb 29e8 c46f 934e  .d..y...i.)..o.N
```

Department of Mathematics
and Computer Science (IMADA)

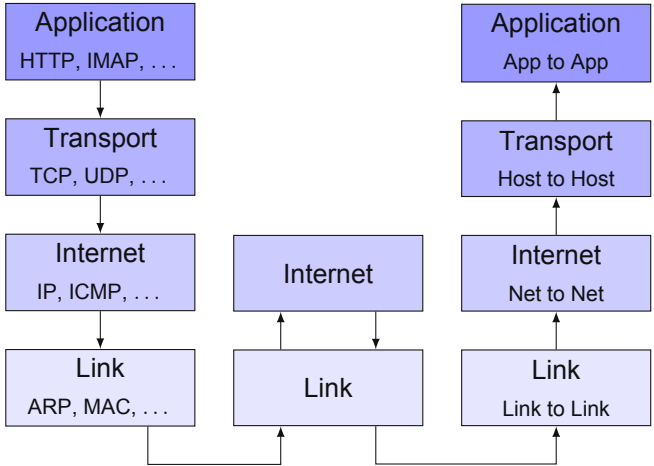
Many modern crypto schemes are using a hybrid-RNG-approach for private keys:

- Use a TRNG to generate a random seed of the desired “strength” (e.g, 128 bit or 256 bit).
- Derive the private key from the seed using the PRNG (often hash-based, e.g., using SHAKE).
- The key can be stored just as the seed; regenerate the private key when needed.
  $\Rightarrow$ Time-memory trade-off.
- An RNG usually is needed also later during encryption or signing for nonces, symmetric keys, IVs, . . .

# Cryptographic Applications

# Cryptography in the Network Layers






S/MIME, PGP, OTR,
TLS, SSH, . . .

IPSec, VPN, TOR, . . .

WPA, WPA2, . . .

# Cryptography beyond Communication



**Data at rest:**

- disc encryption,
- program or data obfuscation,
- . . .

**Data outsourcing:**

- encrypted storage,
- homomorphic encryption,
- . . .

**Authentication:**

- passports,
- NemID,
- biometry,
- . . .

**Rights management:**

- media access,
- feature activation,
- . . .

**Privacy:**

- data mining on anonymized data,
- age verification,
- . . .

**Anonymity:**

- voting systems,
- bidding systems,
- . . .