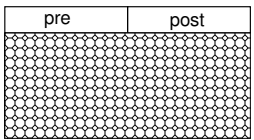
Security by Contract

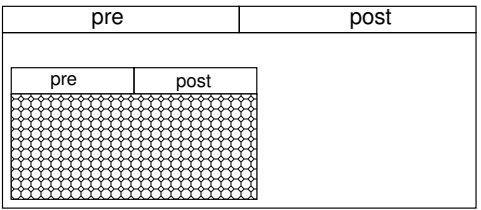
Stefan Lucks

Bauhaus-Universität Weimar

Design by Contract

- Proposed by Bertrand Meyer for re-use of software components
- Communication between user and provider
- Preconditions the user must meet
- Postconditions the provider promises to satisfy

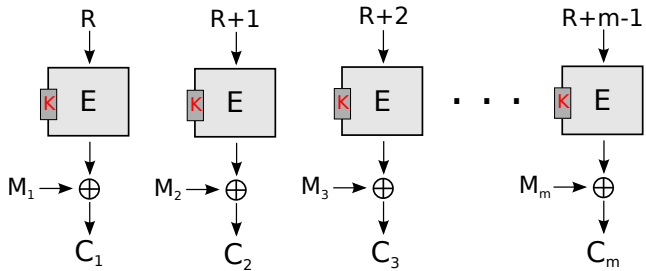




- Nesting: users turn into providers for other users

Example: Counter Mode

Say you just designed it – what contract would you propose?



Block cipher E; random key K
For each msg. $M^i = (M^i_1, \ldots, M^i_{m^i})$:

- choose random R^i
- $C^i_j = E_K(R^i + j - 1) \oplus M^i_j$

Ciphertext $(R, C^i_1, \ldots, C^i_{m^i})$

Contract:
Pre: Assumption: $E_K \approx$ random perm.
Pre: Constraint: $\sigma \ll 2^{n/2}$
Post: Assurance: Ind-CPA

Outlook

Provable Security
Critique
Ideal Primitives
Post-Quantum Security
Contracts for Reduced-Round AES
Example: Faster than Counter Mode
Challenge: Classical Security of Duplex Mode
Summary

Provable Security

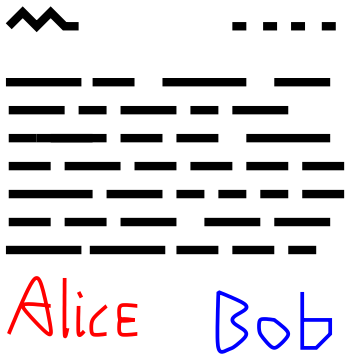
Provable Security

Specify	cryptosystem (“protocol”), composed from primitive(s) (e.g., counter mode, composed from block cipher)	
Define	assumptions on primitive(s) (e.g., secure block cipher)	(pre)
Define	constraints on usage (e.g., data complexity)	(pre)
Define	security assurance (e.g., Ind-CPA)	(post)
Prove	$\boxed{\text{assumptions} \wedge \text{constraints} \longrightarrow \text{assurance}}$	

This is a Contract!

A method of communicating responsibilities

- Two **parties**:
  - provider (cryptosystem designer)
  - user (security engineer)
- **Preconditions** (obligations for user):
  - security of primitives
  - constraints
- **Postconditions** (assurance from provider):
  - security of cryptosystem/protocol



Do we even need the proof?

- Maybe not. But . . .
- Proof is about “enforcing” contract by laws of mathematics.

Historical Landmarks

1979 Rabin: provably secure PK-encryption

1993 Bellare, Rogaway: random oracles

1997 Bellare: practice-oriented provable-security

Rather than prove asymptotic results about the infeasability of breaking a protocol in polynomial time, we present and prove "exact" or "concrete" reductions.

[. . .] what is probably the central step is providing a model and definition, which does not involve proving anything.

Critique

Critique of Provable Security

“If it is provably secure, it probably isn’t.”

– Lars Knudsens’s .sig

Koblitz, Menezes:

- Another Look at ”Provable Security” (2004)
- Another Look at ”Provable Security” II (2006)
- Another Look at Security Definitions (2011)
- Critical Perspectives on Provable Security: Fifteen Years of ”Another Look” Papers (2019):
  → list many publications where “provable security” went practically wrong

Closing Sentences of 2019 Paper

> *[. . .] researchers in "provable security" should strip away unnecessary formalism, jargon, and mathematical terminology from their arguments and strive to make their work "look easy." If they do so, their influence on real-world cryptography will undoubtedly become much greater than it is today.*
>
> – Koblitz, Menezes (2019)

As I would put it,

- a complicated contract doesn't serve its purpose,
- a complicated cryptosystem is difficult to implement correctly, and
- a complicated proof may have hidden flaws.

In short: KISS! ("Keep it simple, stupid!")

Ideal Primitives

The Random Oracle

- Random function $H : \{0,1\}^* \to \{0,1\}^n$
- Useful abstraction for lots of provably security results
- Similar models:
  - ideal (block) cipher, ideal permutation, ...
- Lazy sampling:

  - set $S := \{\}$ (set of pairs $(x, y) \in (\{0,1\}^*, \{0,1\}^n)$)
  - query for $H(X)$:
    - if $\exists Y : (X, Y) \in S$: return Y
    - else choose $Y \in_R \{0,1\}^n$; $S := S \cup \{(X, Y)\}$; return Y

- No efficient implementation!

A Shock!

- Paradoxical result from Canetti, Goldreich, Halevi (1998):
  - a cryptosystem, provably secure in the ROM, but insecure when instantiated with *any* efficient hash function
- Since then, more paradoxical results
- Wide movement towards proofs avoiding random oracles
  - Bellare, Hoan and Keelveedhi (2013) counted 286 papers having *"without random oracles"* in the title

Example “The Jane Doe Protocol”

- Entity recognition: *how do you know several messages from “Jane Doe”, are from the same entity?* (No digital signatures!)
- Lucks, Zenner, Weimersdorf, Westhoff (extended abstract 2005), (full paper 2008):
  - two primitives: hash function h, MAC F
  - to authenticate x_j
    - Bob already knows $\mathbf{a_{j+1}}$
    - Jane Doe sends $y := F_{\mathbf{a_j}}(x_j)$ to Bob
    - ...
    - Jane Doe sends $\mathbf{a_j}$ to Bob;
    - Bob verifes $\mathbf{a_{j+1}} = h(\mathbf{a_j})$ and $F_{\mathbf{a_j}}(x_j) = y$

      $\boxed{a_{j+1} = h(\mathbf{a_j})}$ and $\boxed{F_{\mathbf{a_j}}(x_j) = y}$ $\leftarrow$ (same $\mathbf{a_j}$ used twice)
    - ...

The Evolution of our Proof

0. Almost trivial: both h and F being independent random oracles
1. Proof assuming h to be a random oracle and F a secure MAC
   (proof sketch, but failed to write full proof; eventually counterexample)
2. Standard model proof based on three assumptions
   (correct, but rejected multiple times)
   (1) h secure (2) F secure (3)[1] safe to use key $\mathbf{a_j}$ in both h and F
3. Simple standard model proof (published)
   - assume secure primitve G
   - set $h(k) = G_k(0||\text{const})$, $F_k(x) = G_k(1||x)$

$\rightarrow$ Result 3. is the best!

$\rightarrow$ But in hindsight, I would prefer a correct proof in the ROM over 2.

[1] I guess, (3) would qualify as "bodacious assumption" (Koblitz, Menetzes, 2019).

Discussion

- Strictly mathematical, the random oracle should be abandoned:
  - an efficient hash function H isn't a random oracle (namely, there exists an efficient implementation of H), and
  - from a false assumption, one can derive false conclusions.

- The engineering point of view is different:
  [. . .] it should be noted that no real-world protocol failures have been found that result from the use of random oracles [. . .]
  – Koblitz, Menetzes (2015)

My Take

- Good reasons to avoid random oracles (and other idealized primitives):
  1. mathematical purity (if you care about that)
  2. the random oracle spoils the contract
     - no efficient hash function is a random oracle
     - proof still useful, but contract no longer “enforced” by laws of mathematics
  3. post-quantum security (see below)
- If no standard model proof, or if standard model violates KISS, then go ahead with random oracles (and other idealized primitives)!

Post-Quantum Security

Classical and Quantum Attack Settings

	adversary			**challenger**
classical	$b = f(a)$	$\longrightarrow$	c	$\longrightarrow$
	$\vdots$	$\longleftarrow$	d	$\longleftarrow$
	$x = g(e)$			
quantum Q1	$\lvert b\rangle = \lvert b\rangle \oplus U_f \lvert a\rangle$	$\longrightarrow$	c	$\longrightarrow$
	$\vdots$	$\longleftarrow$	d	$\longleftarrow$
	$\lvert x\rangle = \lvert x\rangle \oplus U_g \lvert e\rangle$			
quantum Q2	$\lvert b\rangle = \lvert b\rangle \oplus U_f \lvert a\rangle$	$\longrightarrow$	$\lvert c\rangle$	$\longrightarrow$
	$\vdots$	$\longleftarrow$	$\lvert d\rangle$	$\longleftarrow$
	$\lvert x\rangle = \lvert x\rangle \oplus U_g \lvert e\rangle$			

The Third Reason

. . . to avoid the random oracles (and other idealized primitives)

- A standard model proof trivially applies to classical and Q1.
- Contract $\boxed{\text{assumptions} \wedge \text{constraints} \longrightarrow \text{assurance}}$ still valid!
- Need to revisit assumptions:

  factorization feasible, longer keys, . . .

- A proof in the random oracle model does not apply to Q1.

What's the Issue with Proofs in the ROM?

- Lazy sampling on classical computers

- set $S := \{\}$ (set of pairs $(x, y) \in (\{0,1\}^*, \{0,1\}^n)$)
- query for $H(X)$:
  - if $\exists Y : (X, Y) \in S$: return Y
  - else choose $Y \in_R \{0,1\}^n$; $S := S \cup \{(X, Y)\}$; return Y

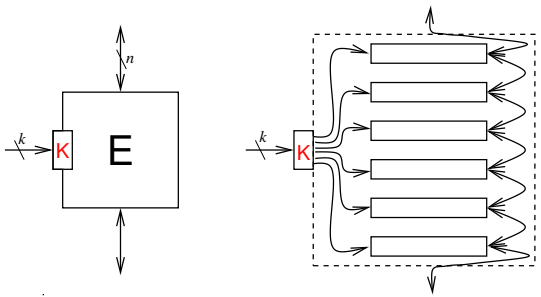
- Quantum computers can't lazily sample (no cloning theorem)!

  $\Rightarrow$ New proof from scratch
  (e.g., use Zhandry's compressed oracle technique)

Contracts for Reduced-Round AES

Contracts for Reduced-Round AES

Blockciphers and other primitives are iterating a “simple” round function many times



- Designer: decide how many rounds are needed
- Cryptanalyst:
  - stepping stone towards attacking full primiitve
  - intuitive understanding of “security margin” of full primitive

Early Attacks on DES

"The lack of progress in the cryptanalysis of the full DES led many researchers to analyse simplified variants of DES, and in particular variants of DES with fewer than 16 rounds."

– Biham, Shamir (1992)

# rounds	data	time ($\log_2$)	
6	1	54	[ChE85]
8	2^{40}	40	[Dav87]
15	2^{52}	52	[BiS91]
16 (full)	2^{47}	37	[BiS92]

[ChE85] Chaum, Evertse 1985 [Dav87] Davies (1987) [BiS91] [BiS91] Biham, Shamir, 1991 [BiS92] Biham, Shamir, 1992

A Selection of Attacks on AES-128

neither related-key nor side-channel

# rounds	data	time ($\log_2$)	
10 (full)	2	126.6	[BKR15]
10 (full)	2^{72}	125.9	[TaW15]
7	2^{97}	99	[DFJ13]
6	$2^{76.6}$	76.6	[Y++24]
6	2^{8}	105.2	[DeF15]
5	$2^{21.5}$	21.5	[B++18]
5	2^{9}	16.5	[D++20]
5	2^{8}	40	[Tun12]
4	4	(negl)	[RBH17]

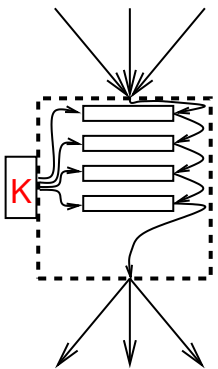
[Tun12] Tunstall (2012) [DeF15] Derbez, Fouque (2015) [TaW15] Tao, Wu (2015) [BKR15] Bogdanov, Khovratovich, Rechberger (2015)
[RBH17] Rønjom, Bardeh, Helleseth (2017) [B++18] Bar-On, Dunkelman, Keller, Ronen, Shamir (2018)
[D++20] Dunkelman, Keller, Ronen, Shamir (2020) [Y++24] Yan, Tan, Xu, Qi (2024)

Reduced number of rounds if cipher is only exposed to very weak attacks?

Let's Write a Contract for 7-round AES (No Proof)!

Full (10-round) AES: assume $\geq$ 125-bit security, even given the entire codebook

Specification E' is 7-round AES, k-bit key, key-schedule from AES-k ($k \in \{128, 192, 256\}$)



Obligations for user:

- key K indistinguishable from random
- constraint: A given at most 16 pairs $(P_i, E'_K(P_i))$; the P_i chosen at random

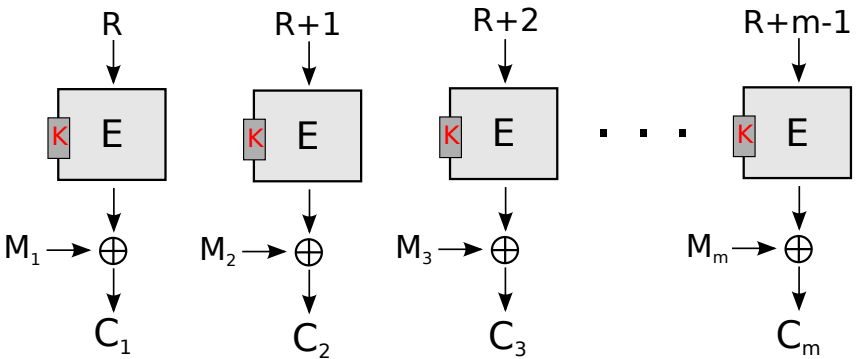
Assurance:

$$\forall A : \frac{\text{time}(A)}{\text{ADV}^{\text{PRP}}_{E'}(A)} \geq 2^{125}$$

Example: Faster than Counter Mode

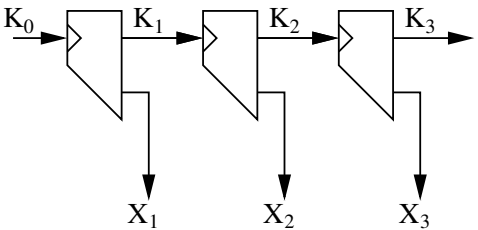
Recall the Counter-Mode Encryption

Is there a way towards faster block cipher based encryption?



- AES-128 (10 rounds/block)
- No contract to for less than 10 rounds!

The Bellare-Yee Generator (2003)



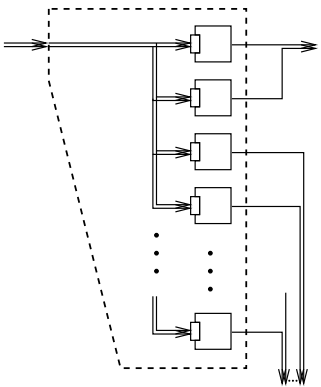
- key K_0
- $(K_i, X_i) := F(K_{i-1})$
- output $X_1, X_2, \ldots$

Assumption: secure PRF $F : \{0,1\}^k \to \{0,1\}^{k+n}$

Constraint: output-length $\sigma \ll n \cdot 2^{k/2}$

Assertion:
- forward-secure
- indistinguishable from random (security $k - \log_2(\sigma/n)$ bit)

Instantiating F via a Block Cipher

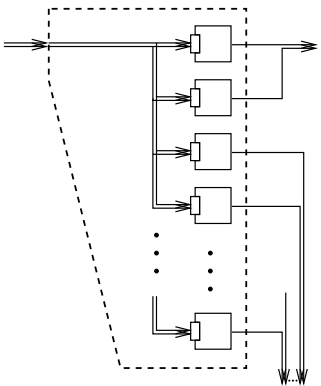


- $n = 128$; $2n$-bit key (for 128 bit security)
- Parameter s
- Public random constants $P_0, \ldots P_{s+1}$
- $F_K : \{0,1\}^{2n} \to \{0,1\}^{2n} \times \{0,1\}^{sn}$
- $K_i := E_{K_{i-1}}(P_0)||E_{K_{i-1}}(P_1)$
- $X_i := E_{K_{i-1}}(P_2)|| \ldots ||E_{K_{i-1}}(P_{s+1})$

- $((E \text{ is secure} \vee s \ll 2^{n/2}) \Leftrightarrow F \text{ is secure})$
  $\Rightarrow$ BY-PRG using F is secure

- Less efficient than running E in counter mode

Instantiating F via a Block Cipher

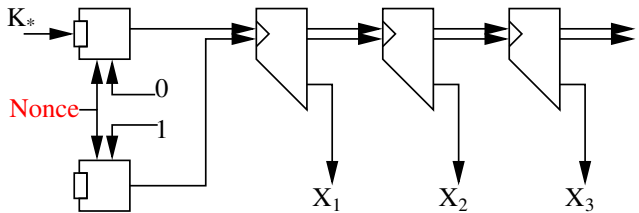


- But: under a given K, E_K called only $s + 2$ times
- For $s + 2 = 16$ we need
  - 7 rounds of AES, but
  - AES-256 key schedule

  ($\rightarrow$ contract above)

- Now $7 * 16$ rounds / 14 blocks
  $=$ 8 rounds / block (Cnt: 10 rounds / block)

Nonce-Based PRG



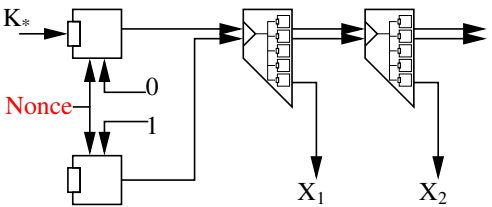
- Two additional block cipher calls to compute 256-bit key

$$K_0 := E_{K_*}(\text{Nonce}||0) \,||\, E_{K_*}(\text{Nonce}||1)$$

from permanent key K_* and 127-bit Nonce

Match and Mix

Match the attack; *Mix* for vastly different attacks.



Nonce processing (2 calls)

- 128 bit key (for $\approx$ 128-bit security)
- unrestricted # plaintexts

$\rightarrow$ full block cipher (10 rounds)

Inside F (16 calls / call to F)

- 256-bit key (still $\approx$ 128-bit security)
- 16 # plaintexts

$\rightarrow$ only 7 rounds

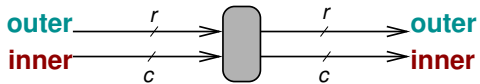
Simplified view at performance:

- if 14-block message: $16 * 7 + 20 = 132$ rounds — 9.4 rounds/block
- for long messages converges to — 8.0 rounds/block

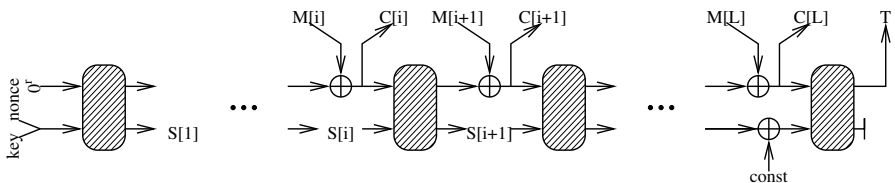
Challenge: Classical Security of Duplex Mode

Recall the Duplex Mode

- $r + c$-bit permutation (r-bit outer state, c-bit inner state)



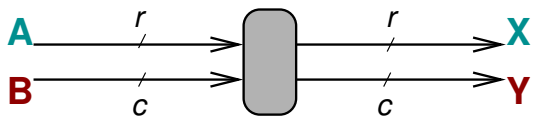
- takes key, nonce, and message $M[1], \ldots M[L]$ (and associated data)
- nonce and key: each $c/2$ bit



- generates ciphertext $C[1], \ldots C[L]$ and authentication tag T

Why Proof in Random Permutation Model?

Why no standard assumption?

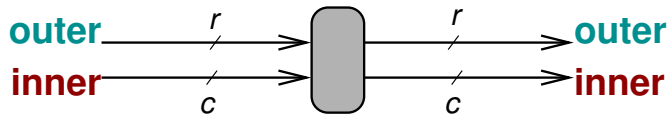


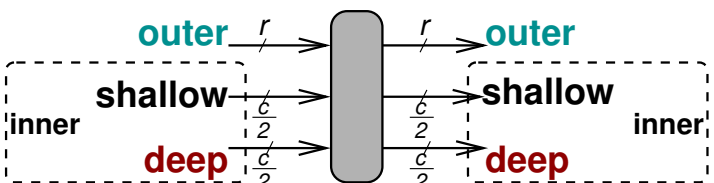
Assume: known **A** secret **B** $\rightarrow$ can't distinguish (**X**, **Y**) from random

Problem: evaluating P^{-1} feasible $\rightarrow$ can distinguish

Revisit Permutation $P : \{0,1\}^{r+c} \rightarrow \{0,1\}^{r+c}$

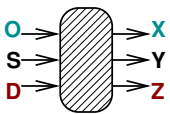
c-bit inner state $\longrightarrow$ ($c/2$-bit middle state, $c/2$-bit innner state)

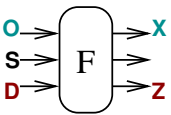


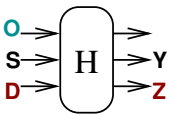


Defining Classical Advantages

Auxiliary functions and advantages for adversary A







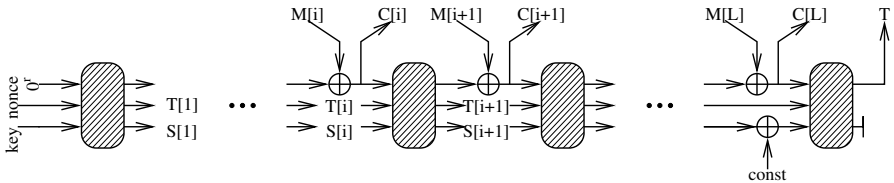
$$F_{\mathbf{D}}(\mathbf{O},\mathbf{S}) = (\mathbf{X},\mathbf{Z}); \qquad H_{\mathbf{D}}(\mathbf{O},\mathbf{S}) = (\mathbf{Y},\mathbf{Z})$$

$$\mathrm{Adv}_P^{\mathrm{PRF}}(A) = |\Pr[A^{F_{\mathbf{D}}} = 1] - \Pr[A^{\$} = 1]|$$

$$\mathrm{Adv}_P^{\mathrm{COLL1}}(A) = \Pr[A^{F_{\mathbf{D}}} = (\mathbf{O},\mathbf{S},\mathbf{O}^*,\mathbf{S}^*) : ((\mathbf{O},\mathbf{S}) \neq (\mathbf{O}^*,\mathbf{S}^*)) \wedge H_{\mathbf{D}}(\mathbf{O},\mathbf{S}) = H_{\mathbf{D}}(\mathbf{O}^*,\mathbf{S}^*)]$$

$$\mathrm{Adv}_P^{\mathrm{COLL2}}(A) = \Pr[A^{F_{\mathbf{D}},F_{\mathbf{D}^*}} = (\mathbf{O},\mathbf{S},\mathbf{O}^*,\mathbf{S}^*) : H_{\mathbf{D}}(\mathbf{O},\mathbf{S}) = H_{\mathbf{D}^*}(\mathbf{O}^*,\mathbf{S}^*)]$$

The Challenge



→ Can we bound security by a function of
- the data complexity σ
- and the advantages $\mathrm{ADV}_P^{\mathrm{PRF}}(A)$, $\mathrm{ADV}_P^{\mathrm{COLL}1}(A)$, and $\mathrm{ADV}_P^{\mathrm{COLL}2}(A)$?

Or do we need different definitions for adversarial advantages?

→ Can we apply "Match and Mix"
(first permutation more rounds, remaining permutations less rounds)?

→ Can we apply this to Duplex variants (e.g., Ascon)?

Summary

Summary

- Similarily to “Design by Contract”, provable security describes
  - the preconditions for using a component, and
  - the assurance (postconditions) provided by that component.

  The proof is useful, but not essential.
- Simplicity is a virtue: KISS!
- There are good reasons to avoid ideal primitives,
  - including post-quantum security,

  but avoid overly complex schemes, bodacious assumptions, etc.
- You can tweak a primitive’s number of rounds, depending on the attacks it is exposed to. You can even “Match and Mix” your primitives.
- Example: block cipher based encryption, faster thn Cnt.

- Challenge: a standard-model analysis for the duplex mode.

One Final Point: We are Hiring!

I am looking for

- a PhD student, or
- a post-doc, who recently finished their PhD.

If you know someone who might be interested, please tell me!

Photo: Ralf Herrmann