Thriving in Between Theory and Practice:
How Applied Cryptography Bridges the Gap

Matilda Backendal & **Miro Haller**

Applied Cryptography Group
ETH Zurich

Computer Science and Engineering
University of California, San Diego

NIST Crypto Reading Club, April 3, 2024

CAW: Cryptographic Applications Workshop

26TH MAY 2024
EUROCRYPT 2024 AFFILIATED EVENT

CRYPTOGRAPHIC
APPLICATIONS
WORKSHOP

1. Formalizing the security of deployed cryptography. PROOFS
2. Constructing cryptographic primitives and systems for practice. DESIGN
3. The industry perspective on deployment and maintenance of cryptography. "PRACTICE"

CAW: Cryptographic Applications Workshop

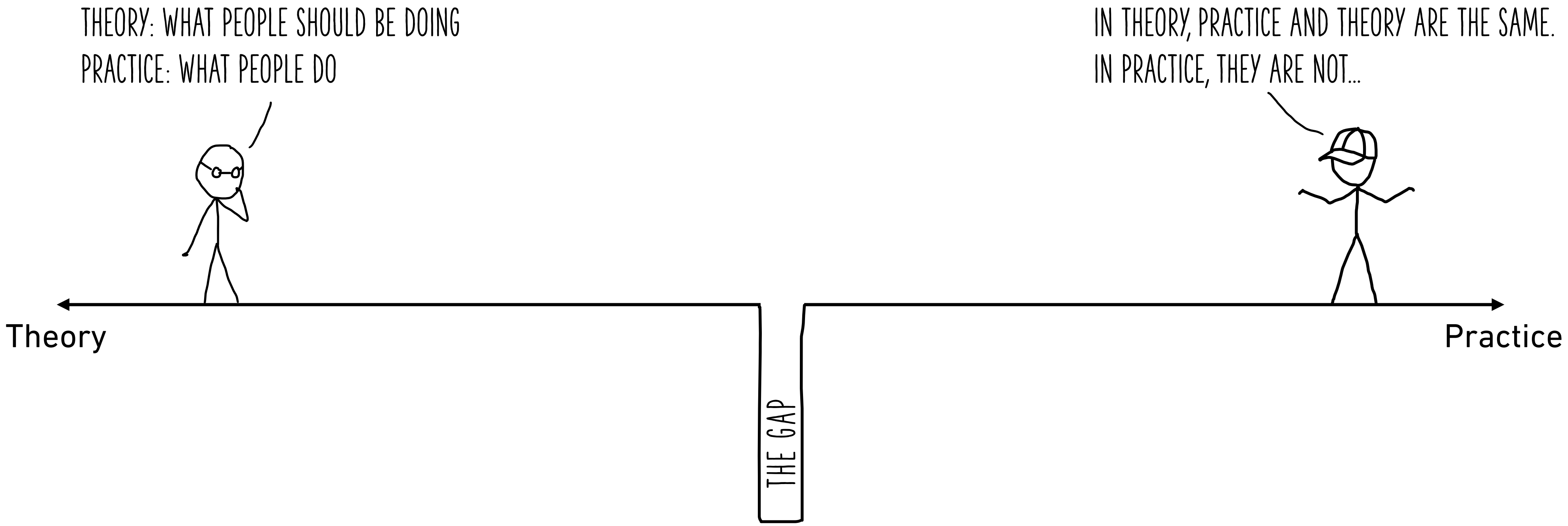
More on caw.cryptanalysis.fun

Sunday, May 26 2024

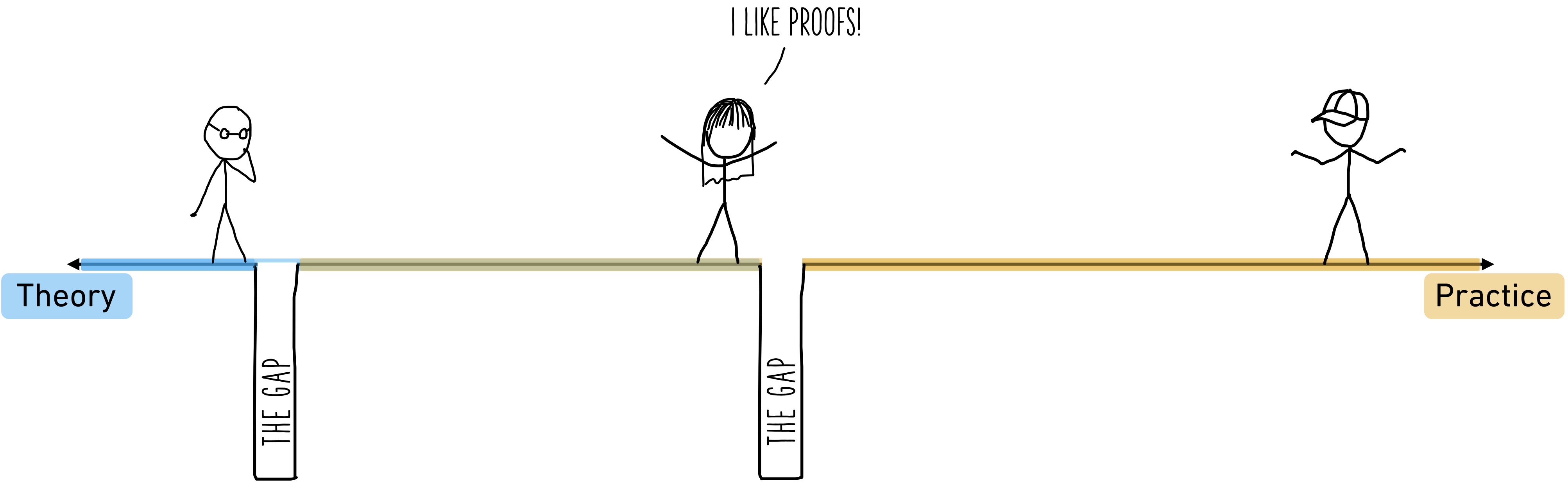
9:10–9:35 (CEST)	"Practical Private Information Retrieval for Real Databases" by **Sofía Celi, Alex Davidson**
9:35–10:00 (CEST)	"How to Encrypt a File at Scale" by **Moreno Ambrosin, Fernando Lobato Meeser**
10:00–10:30 (CEST)	"Analyzing Cryptography in Context: The Case Study of Apple's CSAM Scanning Proposal" by **Gabriel Kaptchuk**
11:00–11:45 (CEST)	"Why we can't have nice (cryptographic) things" by **Henry Corrigan-Gibbs** (invited speaker)
11:45–12:30 (CEST)	"Recent Results on Group Messaging (title TBD)" by **Daniel Collins, Phillip Gajland, Paul Rösler**
13:30–14:00 (CEST)	"Securing semi-open group messaging" by **Fernando Virdia**
14:00–14:30 (CEST)	"A Computational Security Analysis of Signal's PQXDH handshake" by **Rune Fiedler**
14:30–15:00 (CEST)	"Bytes to schlep? Use a FEP: Hiding Protocol Metadata with Fully Encrypted Protocols" by **Aaron Johnson**
15:30–16:00 (CEST)	"Computing on your data with MPC" by **Christopher Patton**
16:00–17:00 (CEST)	Panel on standardization

Standards!

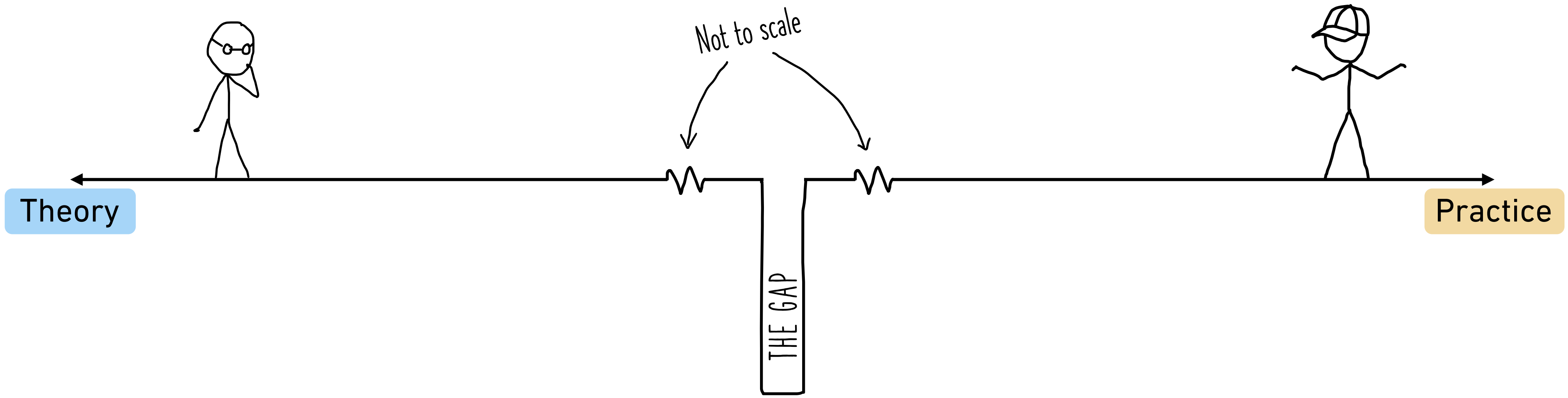
The Gap


THEORY: WHAT PEOPLE SHOULD BE DOING
PRACTICE: WHAT PEOPLE DO
IN THEORY, PRACTICE AND THEORY ARE THE SAME.
IN PRACTICE, THEY ARE NOT...
Theory
Practice
THE GAP

The Gap



The Gap



Taxonomy of Cryptography

	Theory	Practice
Theory	***T2T***	***T2P***
Practice	***P2T***	***P2P***

Bridging the Gap

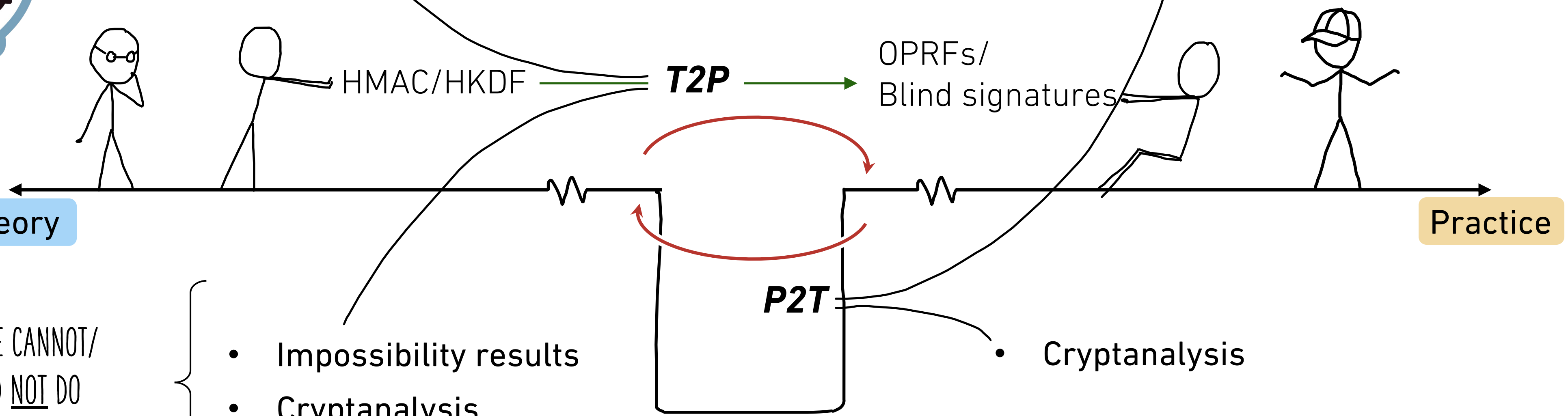
WHAT WE CAN/SHOULD DO

T2P

- Primitive/protocol/scheme design
- Standardization
- ...

P2T

- Provable security
- Formal verification
- ...



- Cryptanalysis

WHAT WE CANNOT/
SHOULD NOT DO

./ WAC

- Impossibility results
- Cryptanalysis

Workshop on Attacks in Cryptography

Dual-PRF Security of HMAC

Based on work with Mihir Bellare, Felix Günther & Matteo Scarlata

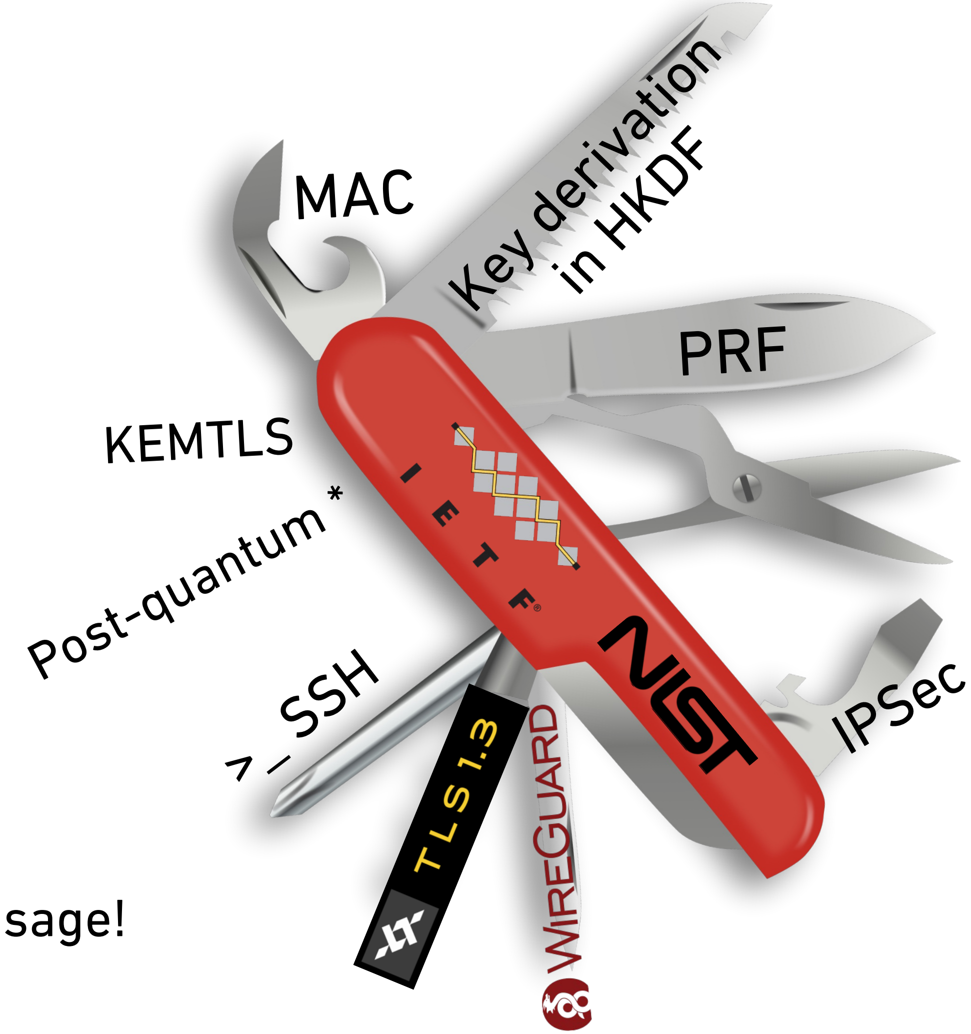
HMAC: the Swiss Army Knife of Crypto

HMAC [CRYPTO'96:BCK] is

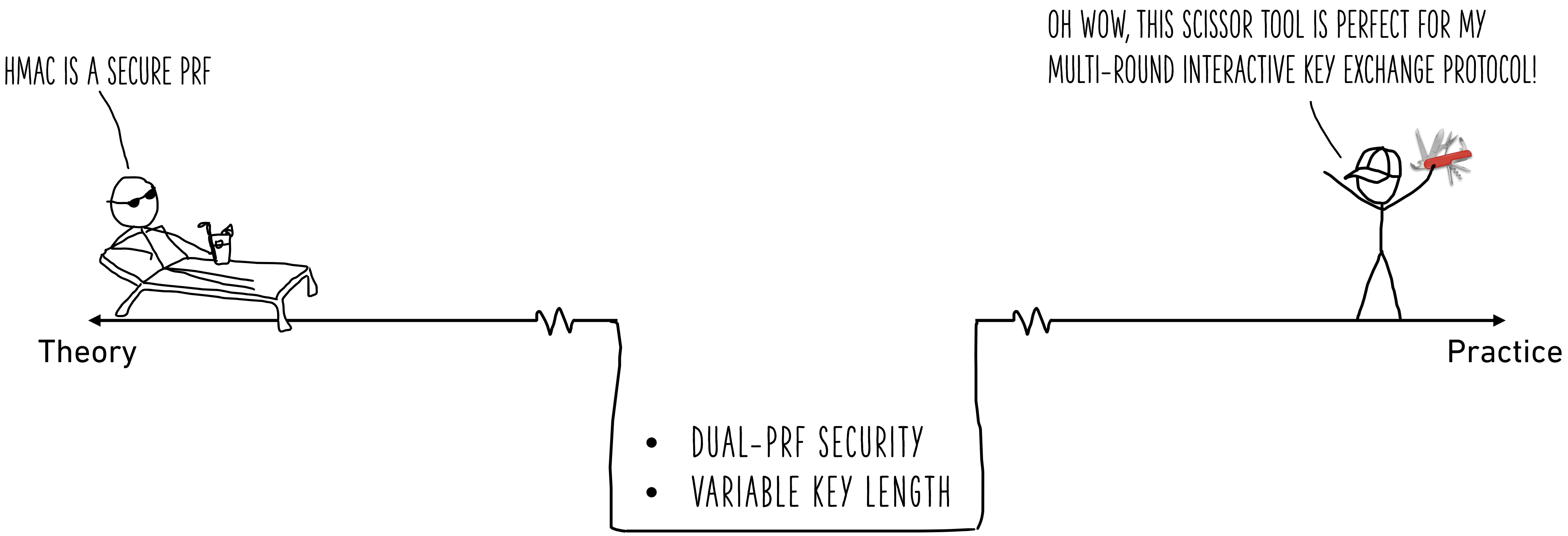
- a hash-based MAC,
  - standardized,
  - provably secure,
  - versatile,
  - and widely used.

...as a PRF
[C'96:BCK, C'06:Bel, C'14:GPR].

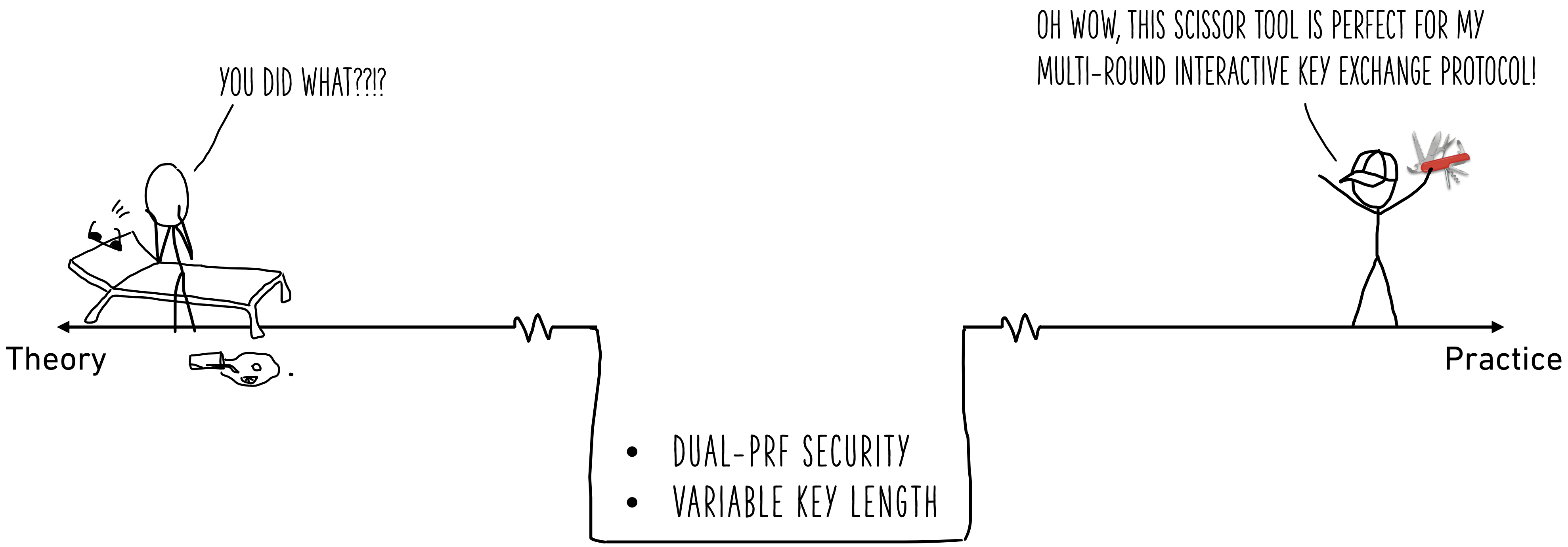
This doesn't match current usage!



The HMAC Gap


HMAC IS A SECURE PRF
OH WOW, THIS SCISSOR TOOL IS PERFECT FOR MY MULTI-ROUND INTERACTIVE KEY EXCHANGE PROTOCOL!
Theory
Practice
• DUAL-PRF SECURITY
• VARIABLE KEY LENGTH

The HMAC Gap

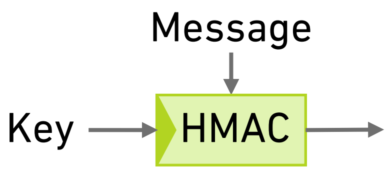

YOU DID WHAT??!?
OH WOW, THIS SCISSOR TOOL IS PERFECT FOR MY MULTI-ROUND INTERACTIVE KEY EXCHANGE PROTOCOL!
Theory
Practice
• DUAL-PRF SECURITY
• VARIABLE KEY LENGTH

HMAC in Action

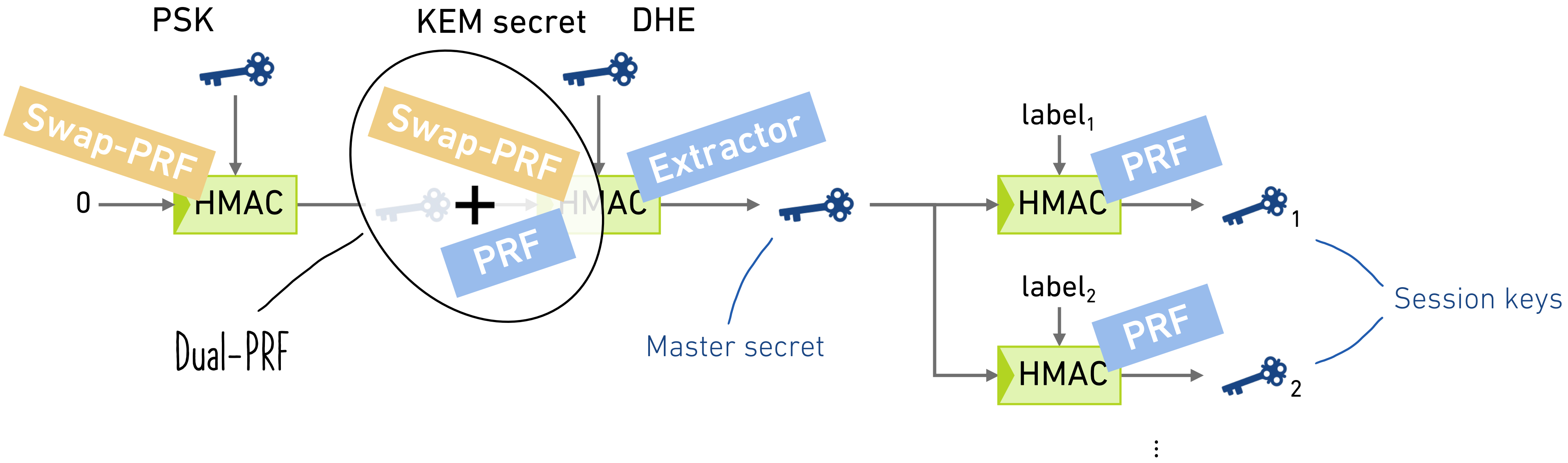
HMAC: $\{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^c$

Key space — Message space

Variable key length



TLS 1.3 Key Schedule



HMAC Is Assumed to Be a Dual-PRF

In the analysis of:

- TLS 1.3 PSK [JoC'22:DFGS]
- KEMTLS [CCS'20:SSW]
- PQ Wireguard [S&P'21:HNSWZ]
- PQ Noise [CCS'22:ADHSW]
- Messaging Layer Security (MLS) [S&P'22:BCK]

THEOREM 4.1. *Let $\mathcal{A}$ be an algorithm, and let n_s be the number of sessions and n_u be the number of parties. Then the advantage of $\mathcal{A}$ in breaking the multi-stage security of KEMTLS is upper-bounded by*

$$\frac{n_s^2}{2^{|\text{nonce}|}} \ldots n_s \left(\epsilon^{\text{IND-1CCA}}_{\text{KEM}_e} + \epsilon^{\text{PRF-sec}}_{\text{HKDF.Ext}} + 2\,\epsilon^{\text{dual-PRF-sec}}_{\text{HKDF.Ext}} + 3\,\epsilon^{\text{PRF-sec}}_{\text{HKDF.Exp}} \ldots \right)$$

The first assumption is concerned with the use of HMAC as a dual PRF (cf. [Bel…

[...]

Theorem 6.2 (Multi-Stage security of TLS1.3-PSK-0RTT). *The TLS 1.3 PSK 0-RTT … is* Multi-Stage*-secure with properties* (M, AUTH, FS, USE, REPLAY) *given above. Formally, for any efficient adversary $\mathcal{A}$ against the* Multi-Stage *security there exist efficient algorithms $\mathcal{B}_1, \ldots, \mathcal{B}_8$ such that*

$$\ldots \mathsf{Adv}^{\text{dual-PRF-sec}}_{\ldots} + \mathsf{Adv}^{\text{PRF-sec}}_{\ldots} \ldots$$

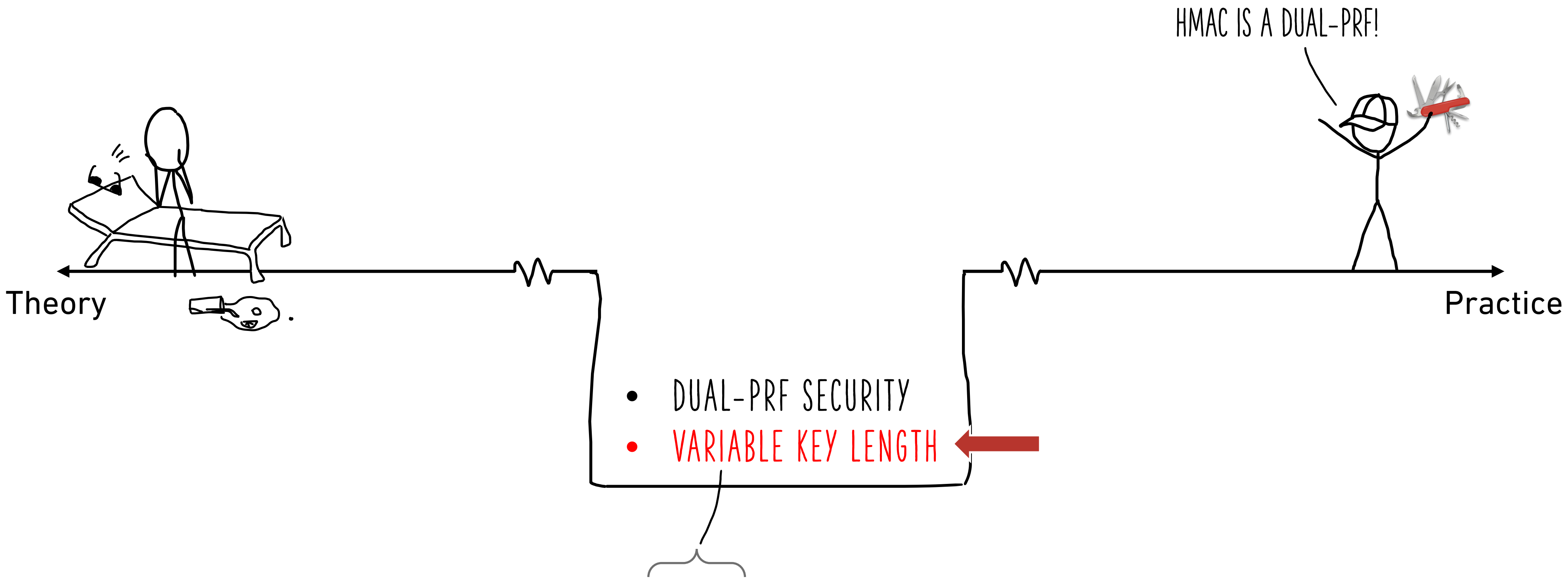
Theorem 1. *A Noise Hash Object NHO is a secure pseudo-random Hash-Object if* HMAC-HASH *is a dual-prf with:* $\mathsf{Adv}^{PRHO}_{NHO,\mathcal{A},q_i}(1^\lambda) \leq \left(\mathsf{Adv}^{\mathsf{CollRes}}_{\text{HMAC-HASH},\mathcal{A}'}(1^\lambda) + \mathsf{Adv}^{\mathsf{PRF\text{-}SWAP}}_{\text{HMAC-HASH},\mathcal{A}'}(1^\lambda) + (2\cdot q)\cdot \mathsf{Adv}^{\mathsf{PRF}}_{\text{HMAC-HASH},\mathcal{A}'}(1^\lambda) \right)$ *where q refers to the total number of oracle-queries.*

…o Appendix A for a proof. Intuitively the …tance of HMAC-HASH implies that only …ries result in equal states and the HMAC-… a dual-PRF (see Appendix B.2) ensures that …en added to a chain, its first state becomes …n which is retained upon subsequent calls.

In PQ-WireGuard a dual-PRF appears in the form of a *key-derivation function* $\mathsf{KDF}(X,Y) = Z$ that takes two inputs, X … and Y, and outputs a bit string Z consisting of three blocks … $Z = Z_1 \| Z_2 \| Z_3$. We write $\mathsf{KDF}_i(X,Y)$ for the i-th block of … output of $\mathsf{KDF}(X,Y)$, i.e., Z_i. The reason why KDF has to … be a dual-PRF is discussed in Section IV-A.

Assumptions. We make standard key indistinguishability and collision-resistance assumptions on the key derivation functions (KDF) and assume indistinguishability under chosen-ciphertext attacks (IND-CCA) secure public-key encryption, as well as that the Extract function in Krawczyk's HKDF design [24] is a dual pseudorandom function and thus, we assume that HKDF is a dual KDF, which has also been assumed in the analysis of Noise [21] and TLS 1.3 [12].

The HMAC Gap



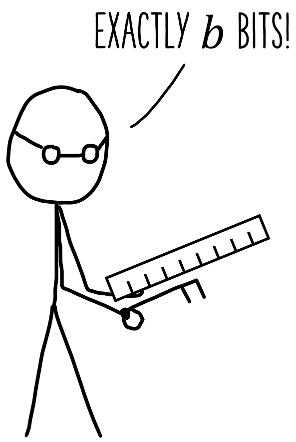
HMAC: $\{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^c$

HMAC in Action

HMAC(K, M) = H((K ⊕ opad) || H((K ⊕ ipad) || M))

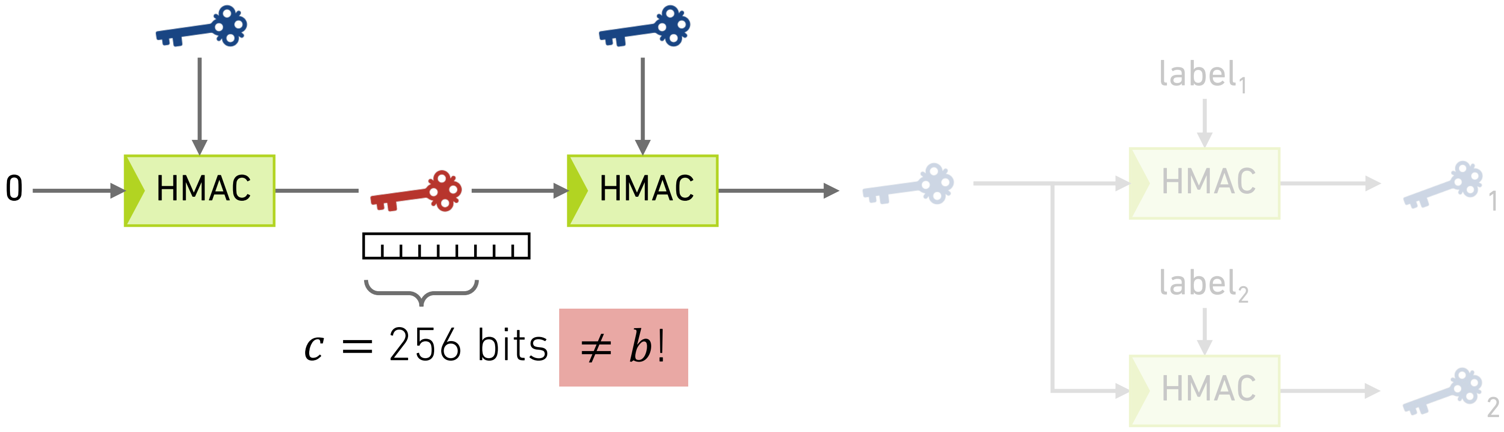
Merkle-Damgård hash function, b-bit constants

e.g. **SHA-256**: $c = 256$, $b = 512$



PRF proof:
$\mathrm{HMAC}_b(K_b, M)$

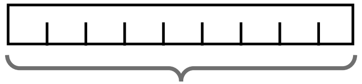
TLS 1.3 Key Schedule



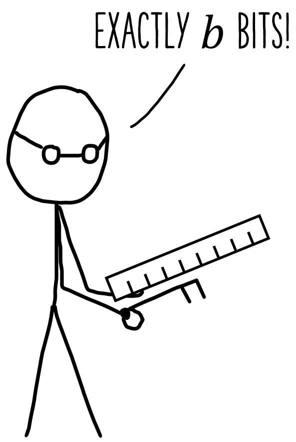
HMAC in Action

$\text{HMAC}(K, M) = H((K \oplus \text{opad}) \,||\, H((K \oplus \text{ipad}) \,||\, M))$

Pad-or-Hash(K)

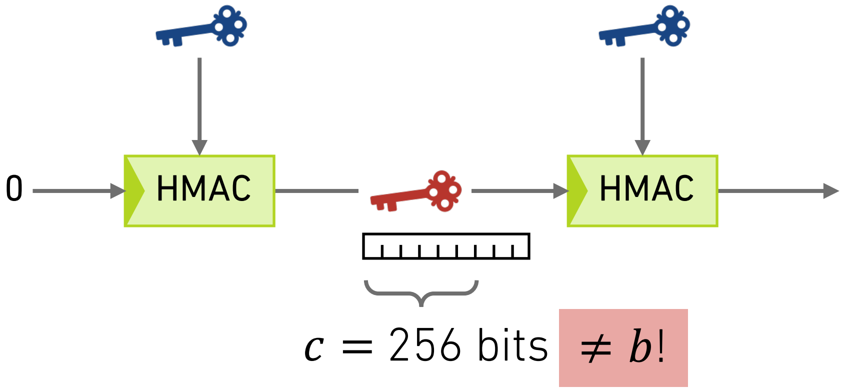


b bits



PRF proof:
$\text{HMAC}_b(K_b, M)$

TLS 1.3 Key Schedule



Summary

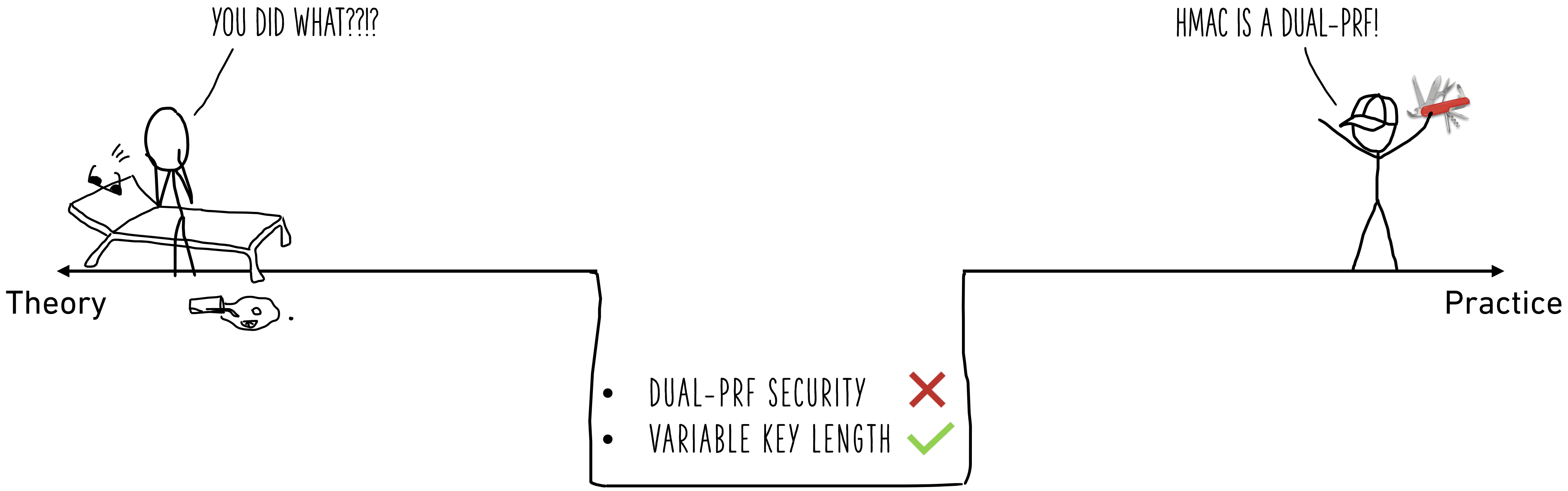
Proof existed:

- ✓ $\text{HMAC}_b(K_b, M)$

No proof existed:

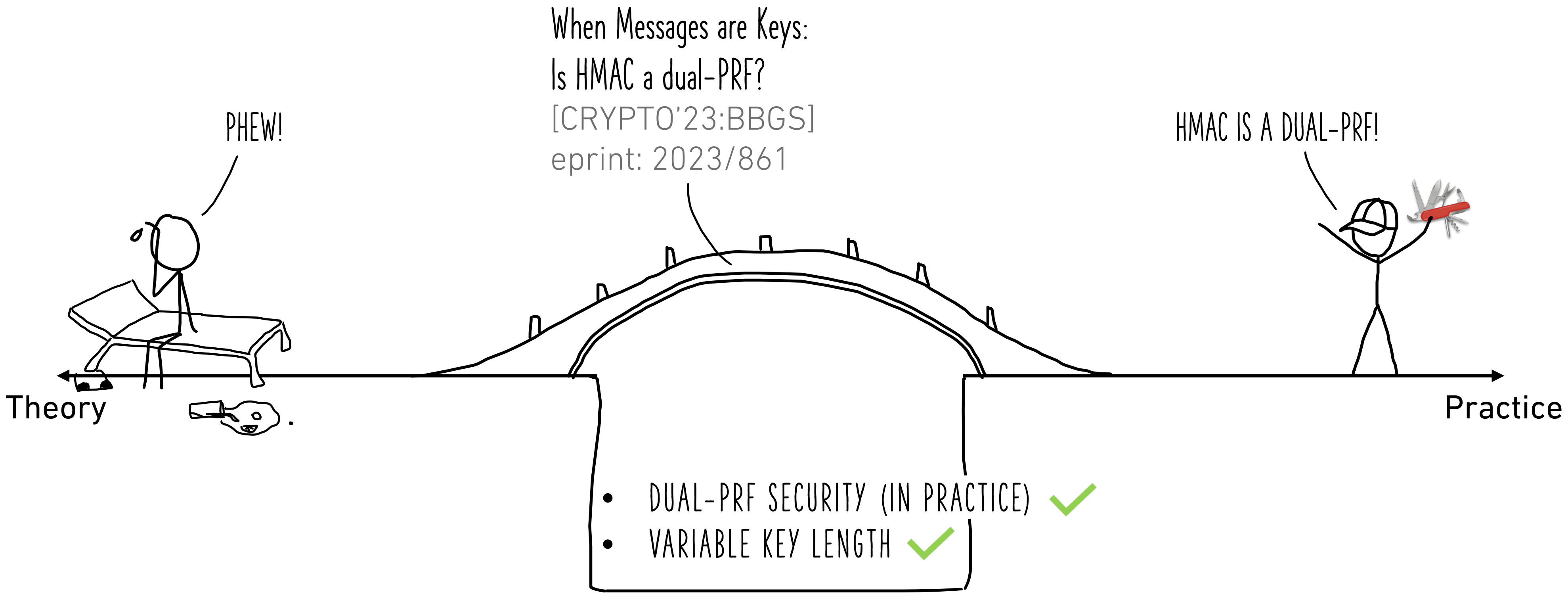
- × $\text{HMAC}(K, M) = H((\text{PoH}(K) \oplus \text{opad}) \,||\, H((\text{PoH}(K) \oplus \text{ipad}) \,||\, M))$
- × $\text{HMAC}(M, K)$

The HMAC Gap



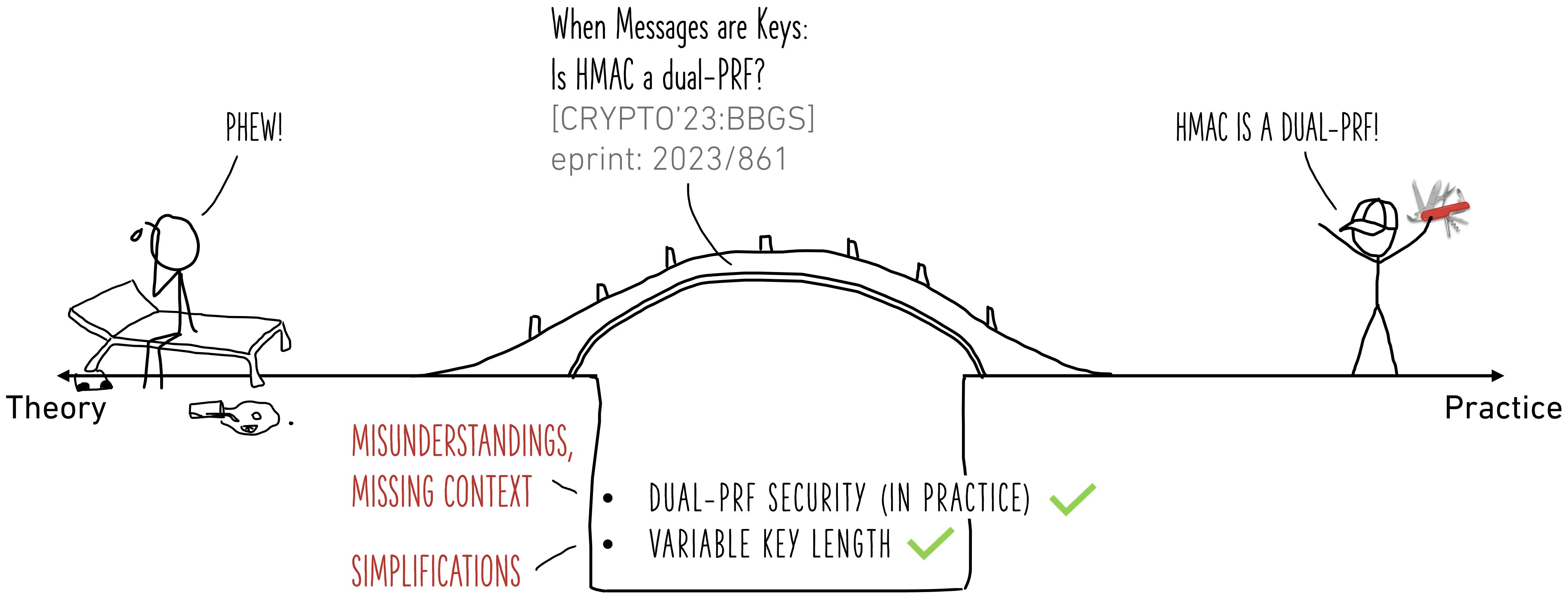
Is HMAC a Variable-Key Length Dual-PRF?

The HMAC Gap



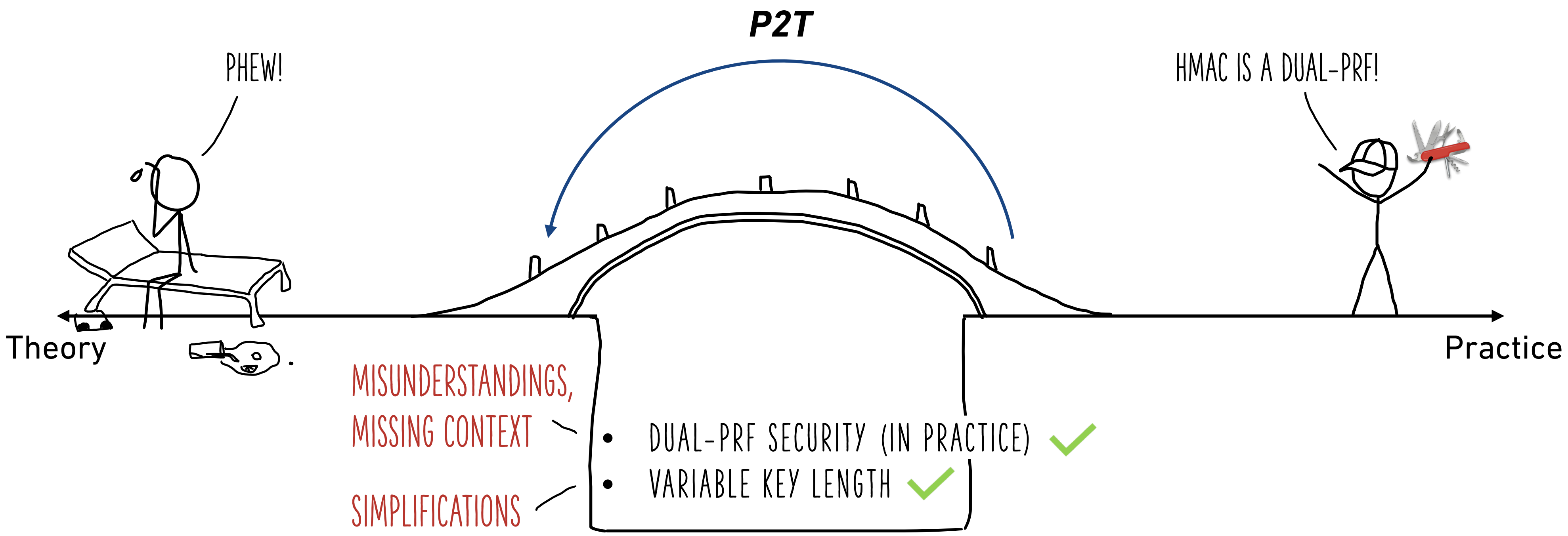
Is HMAC a Variable-Key Length Dual-PRF?

The HMAC Gap



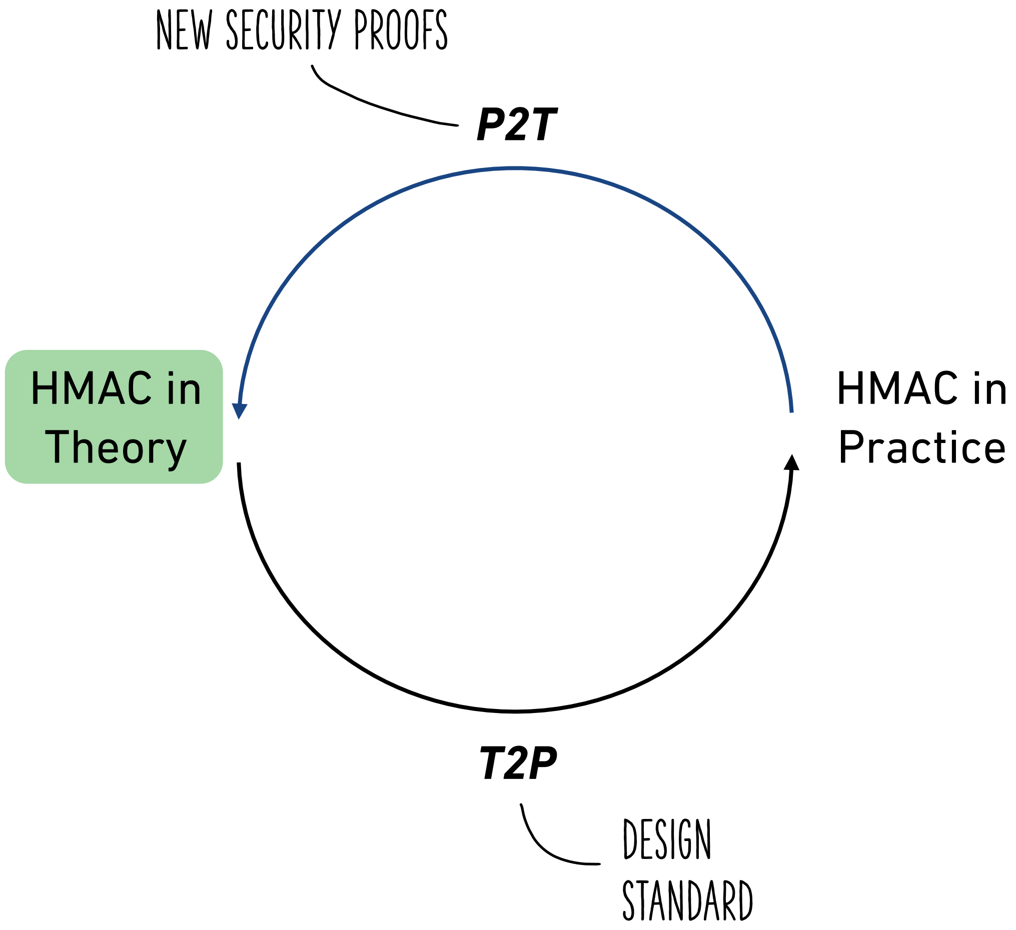
Why Did the Gap Arise?

The HMAC Gap



Why Did the Gap Arise?

The HMAC Cycle



End-to-End Encrypted Cloud Storage

Based on work with Hannah Davis, Felix Günther & Kenny Paterson

Why Do We Want E2EE Cloud Storage?

Privacy

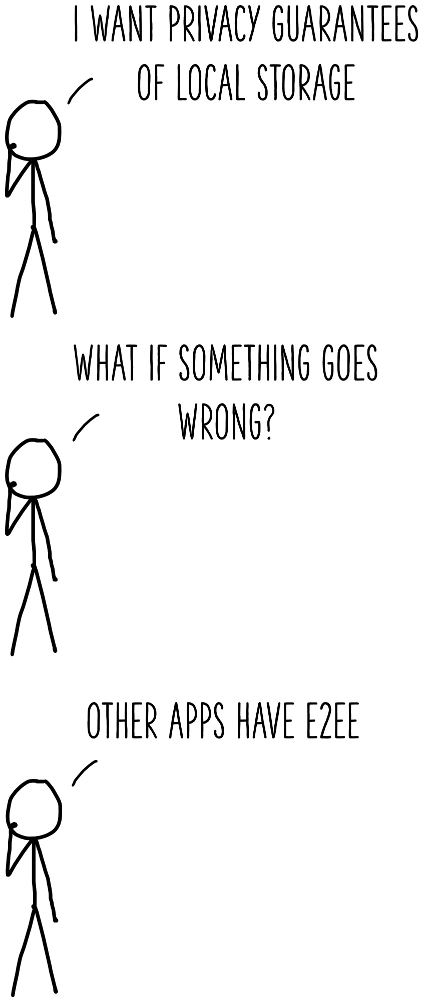
- Sensitive files
- No analytics or data processing

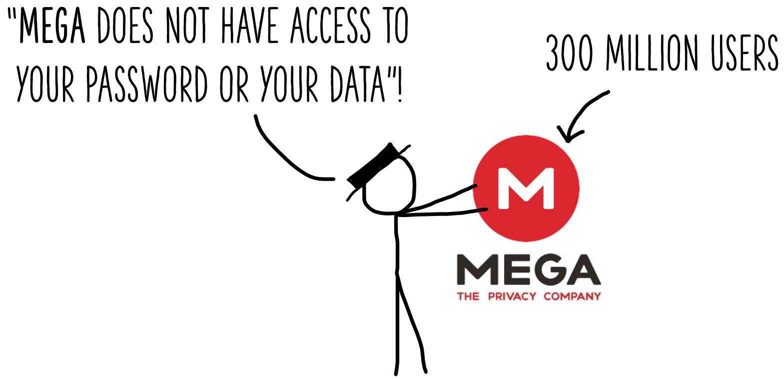
Security

- Untrusted or compromised provider
- Legally compelled to disclose

E2EE in other domains

- Data in transit (browsing, messaging)
- Data at rest (local storage, backups)





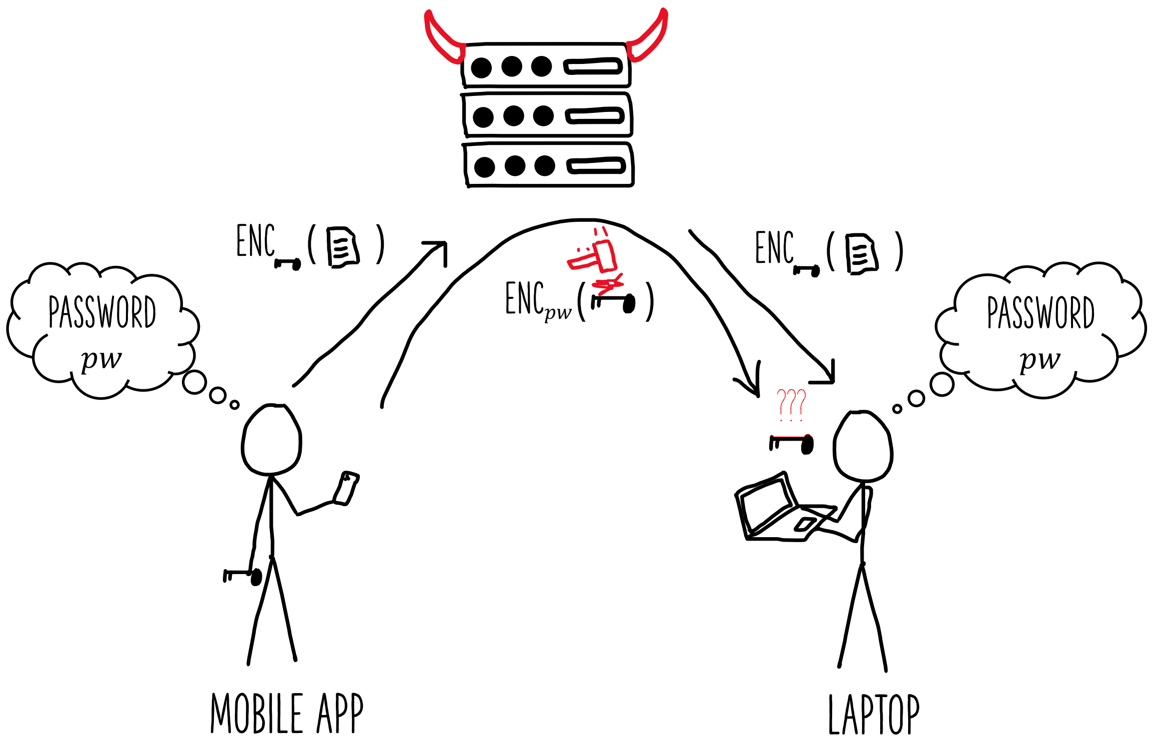
no E2EE per default

- OneDrive
- Dropbox
- Google Drive
- iCloud Drive

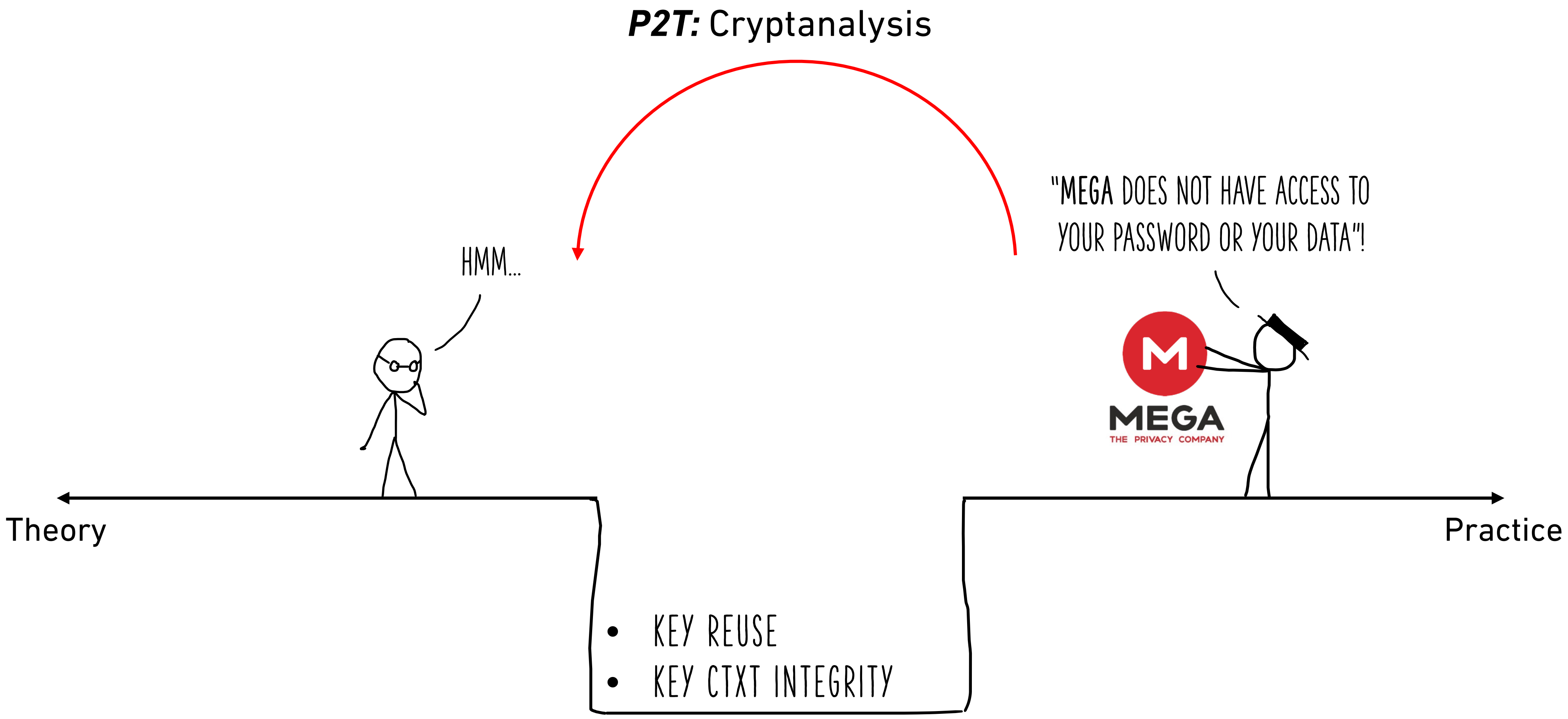
OPTIONAL E2EE AT COST OF FUNCTIONALITY

E2EE Cloud Storage Implementation

- Client-side encryption
  - Pick fresh key to encrypt file
- Issue on download
  - Retrieving key on another device
- Solution
  - Send key encrypted with password over server
- Untrusted server
  - Key overwriting attacks



P2T Example: The Cryptanalysis of MEGA



MEGA: Exploiting Authentication for File Decryption*

*highly simplified

Challenge-response authentication

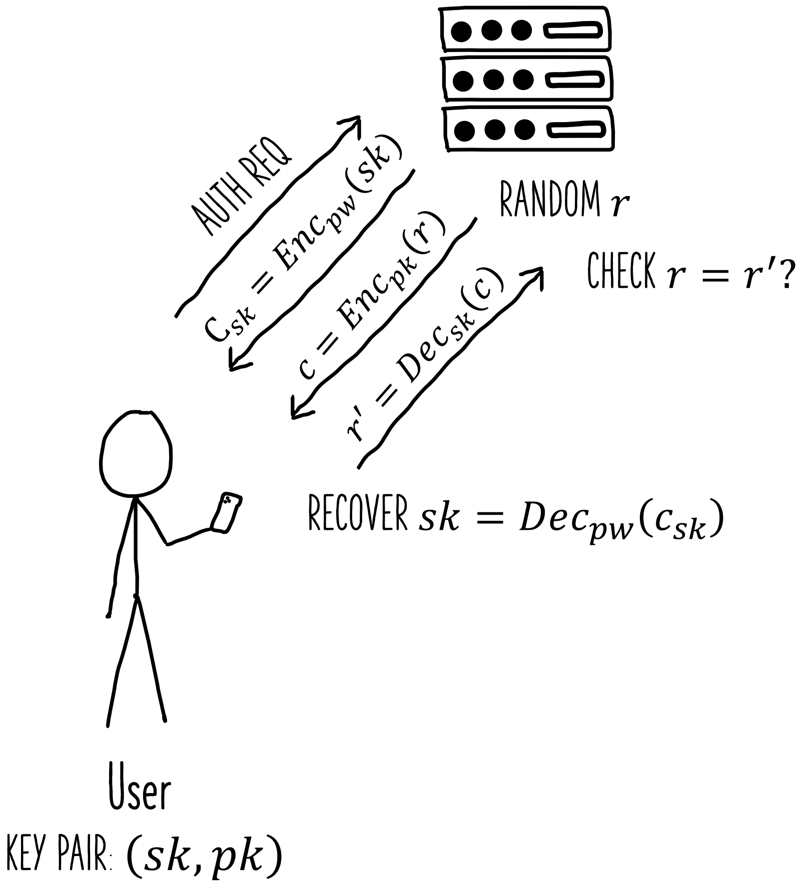
Server:

- Send secret key sk encrypted with password pw
- Encrypt challenge r with user public key pk

User:

- Decrypt secret key ciphertext c_{sk} with pw
- Decrypt challenge c, send recovered r' back

Authentication successful if $r = r'$

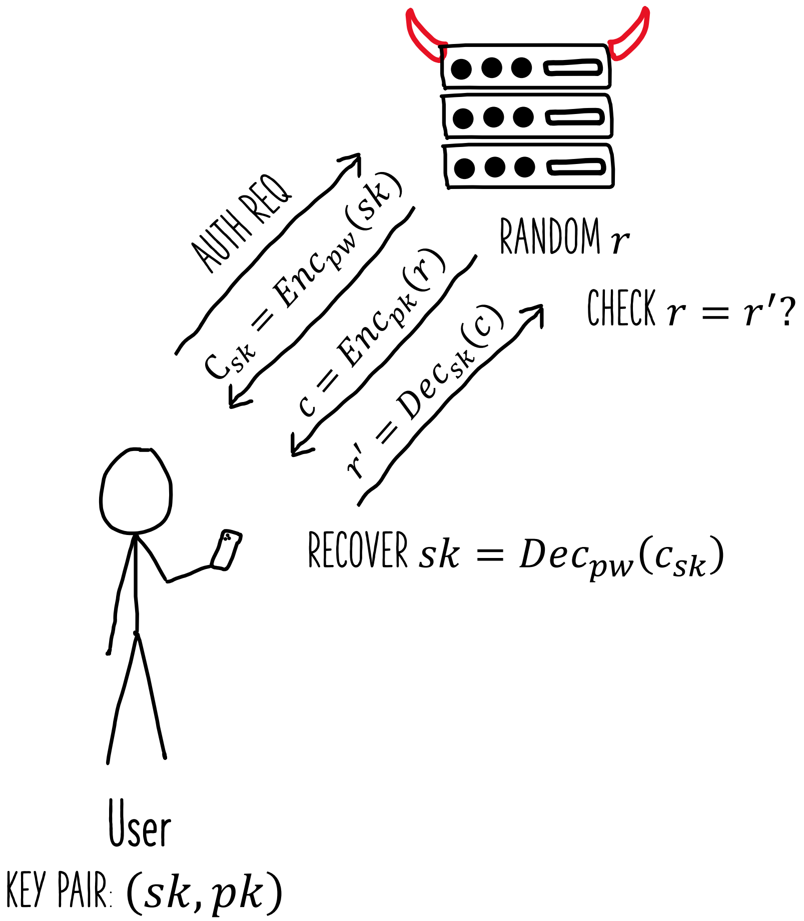


MEGA: Exploiting Authentication for File Decryption*

*highly simplified

Attack

1. [2] attack to recover file keys fk
2. Key reuse: $Enc_{pw}(sk)$ and $Enc_{pw}(fk)$



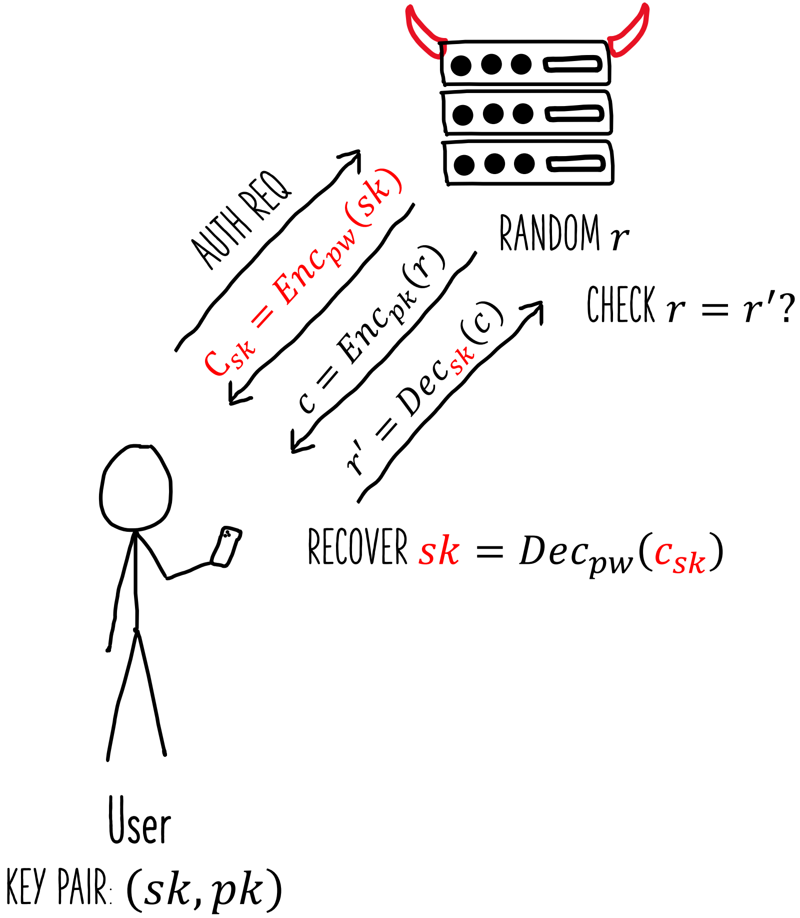
[2] Matilda Backendal, Miro Haller and Kenneth G. Paterson. "MEGA: Malleable Encryption Goes Awry". IEEE S&P 2023.

MEGA: Exploiting Authentication for File Decryption*

*highly simplified

Attack

1. [2] attack to recover file keys fk
2. Key reuse: $Enc_{pw}(sk)$ and $Enc_{pw}(fk)$
3. Partially overwrite c_{sk} with $Enc_{pw}(fk)$
   - No integrity protection of c_{sk}!

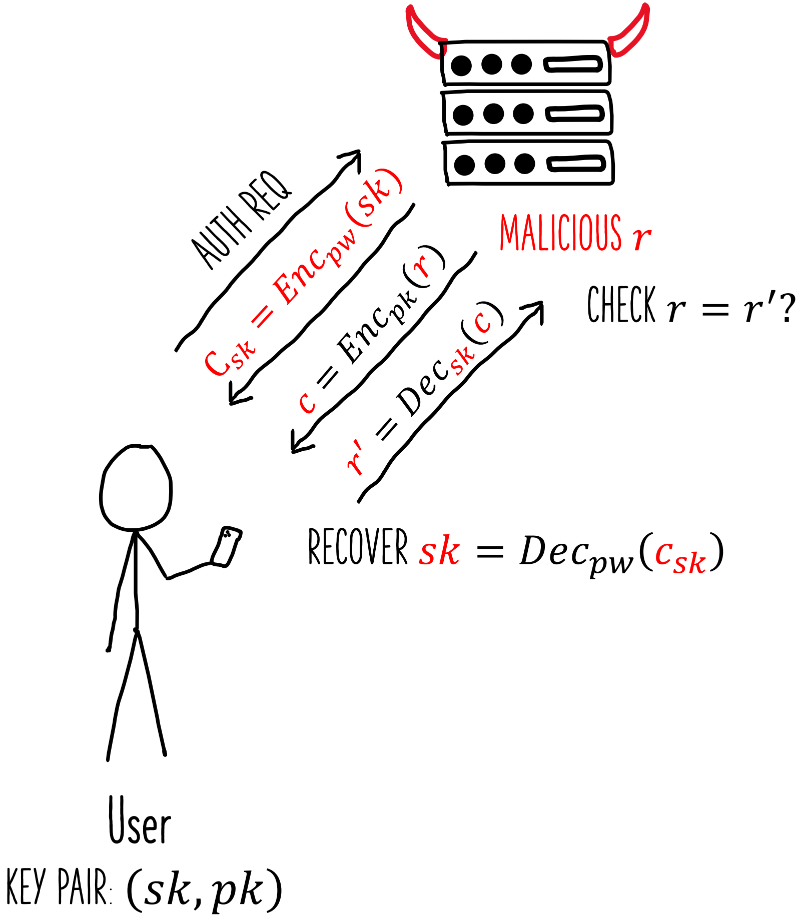


[2] Matilda Backendal, Miro Haller and Kenneth G. Paterson. “MEGA: Malleable Encryption Goes Awry”. IEEE S&P 2023.

MEGA: Exploiting Authentication for File Decryption* *highly simplified

Attack

1. [2] attack to recover file keys fk
2. Key reuse: $Enc_{pw}(sk)$ and $Enc_{pw}(fk)$
3. Partially overwrite c_{sk} with $Enc_{pw}(fk)$
   - No integrity protection of c_{sk}!
4. Pick malicious r



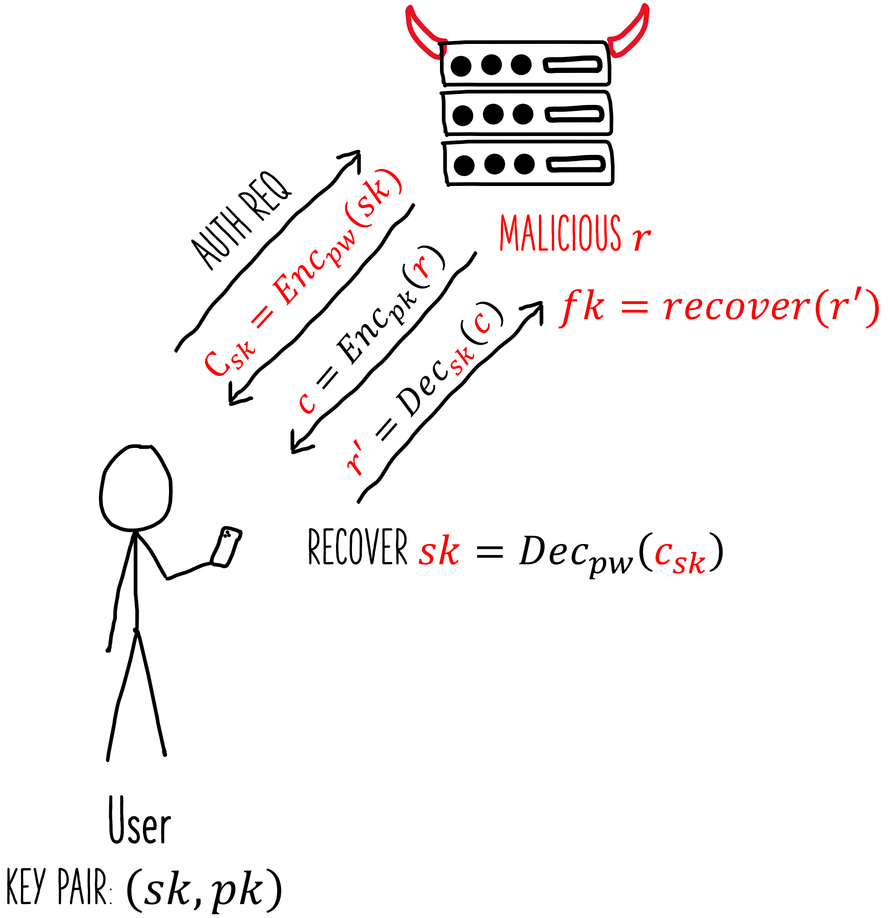
[2] Matilda Backendal, Miro Haller and Kenneth G. Paterson. "MEGA: Malleable Encryption Goes Awry". IEEE S&P 2023.

MEGA: Exploiting Authentication for File Decryption*

*highly simplified

Attack

1. [2] attack to recover file keys fk
2. Key reuse: $Enc_{pw}(sk)$ and $Enc_{pw}(fk)$
3. Partially overwrite c_{sk} with $Enc_{pw}(fk)$
   - No integrity protection of c_{sk}!
4. Pick malicious r
5. Recover fk from r'



[2] Matilda Backendal, Miro Haller and Kenneth G. Paterson. “MEGA: Malleable Encryption Goes Awry”. IEEE S&P 2023.

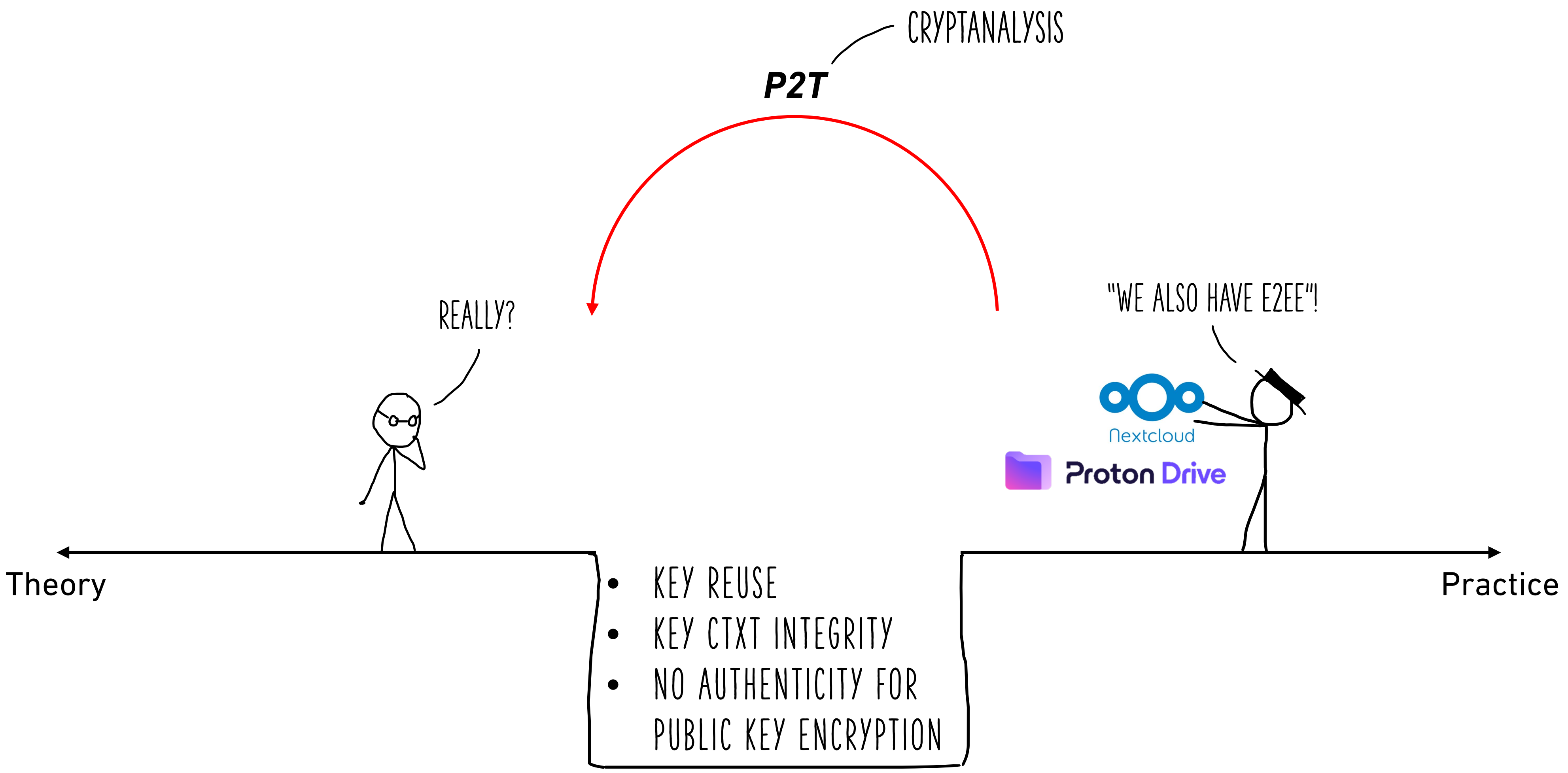
Challenges & Issues in MEGA

- Integrity for key ciphertexts
- Key reuse
- Patching is hard
  - Re-encryption requires > 185 days
- Multi-device access
- Sharing is tricky

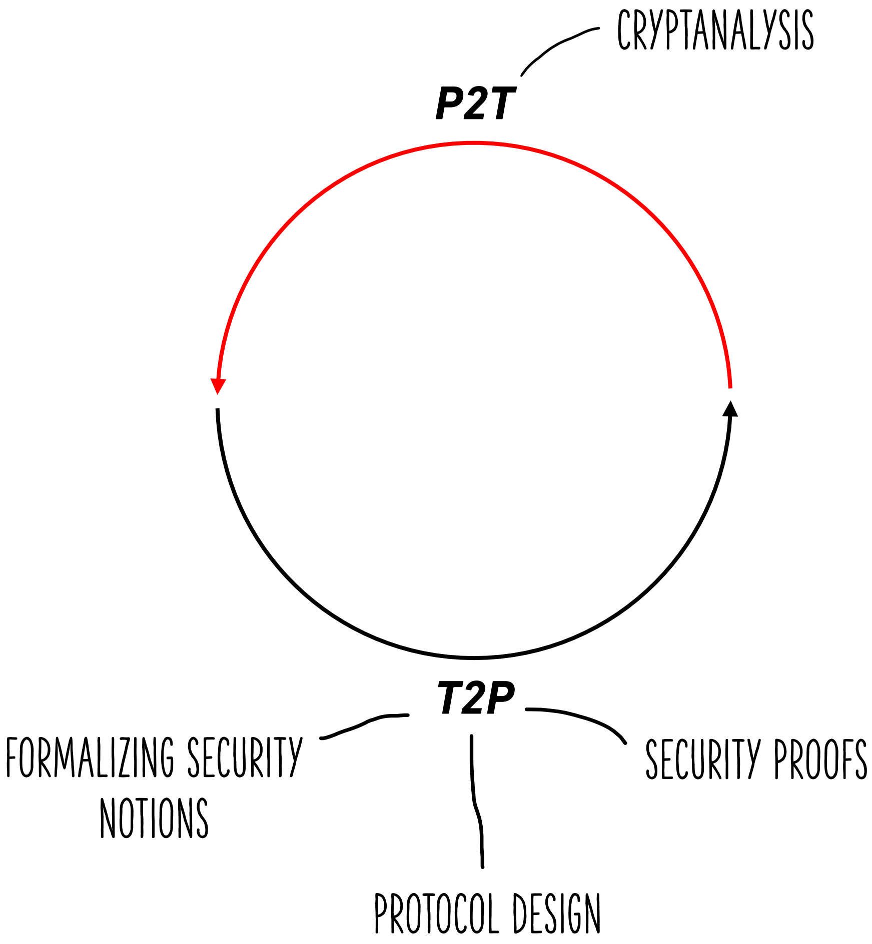
Lessons Learned

- Unclear security goals
- Key separation is essential
- Cryptographic agility & minimize chance of vulnerabilities
- Password-based security
- Interaction with (potentially malicious) users/server

The E2EE Cloud Storage Cycle



The E2EE Cloud Storage Cycle



Security Notions for E2EE Cloud Storage: Operations and Syntax

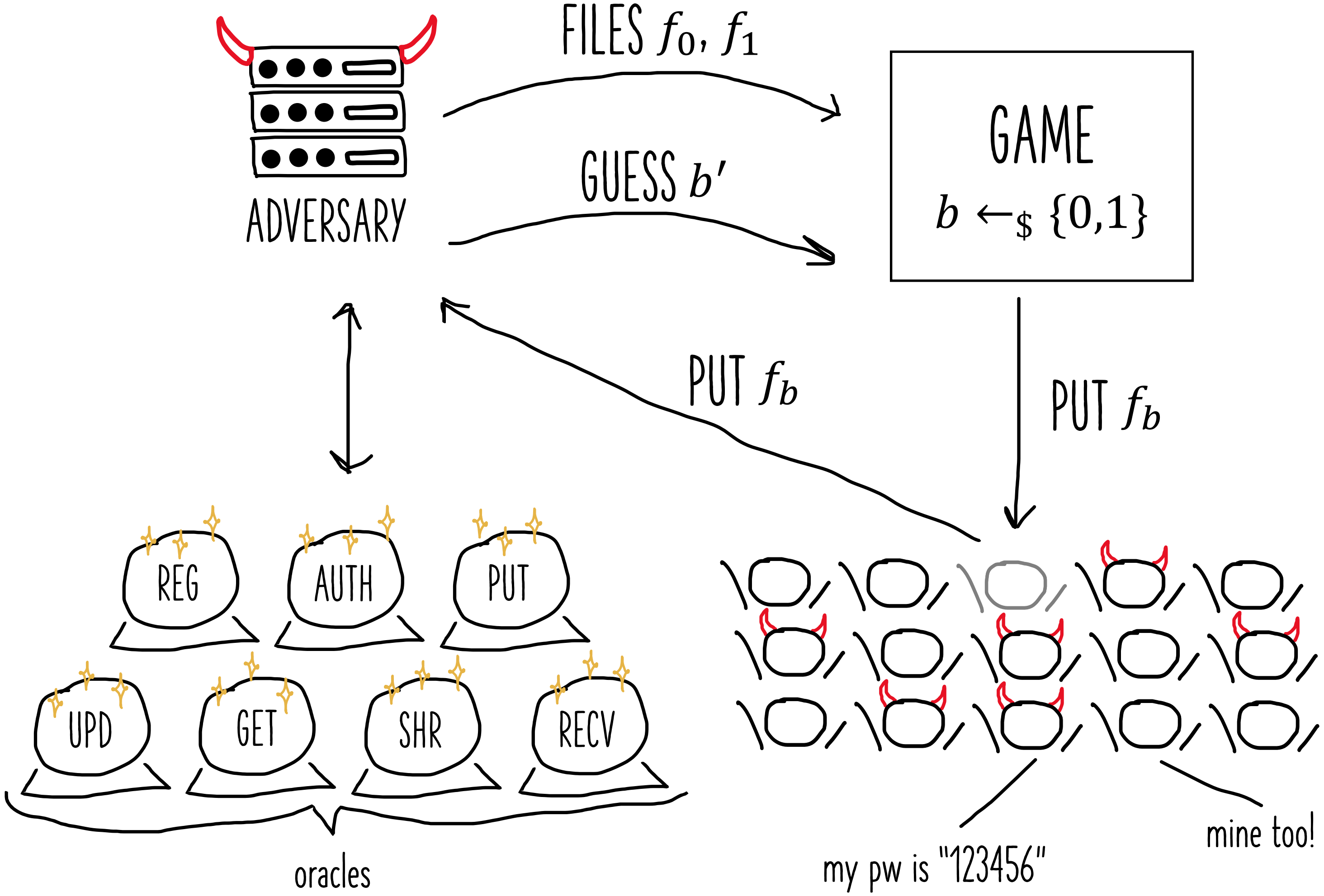
- Identify core functionalities
  - Register (reg)
  - Authenticate (auth)
  - Upload (put)
  - Update (upd)
  - Download (get)
  - Share (shr)
  - Receive (recv)
- Define syntax to express them
  - Non-atomic operations
  - Allow arbitrary interleavings

MEGA
Nextcloud
Proton Drive

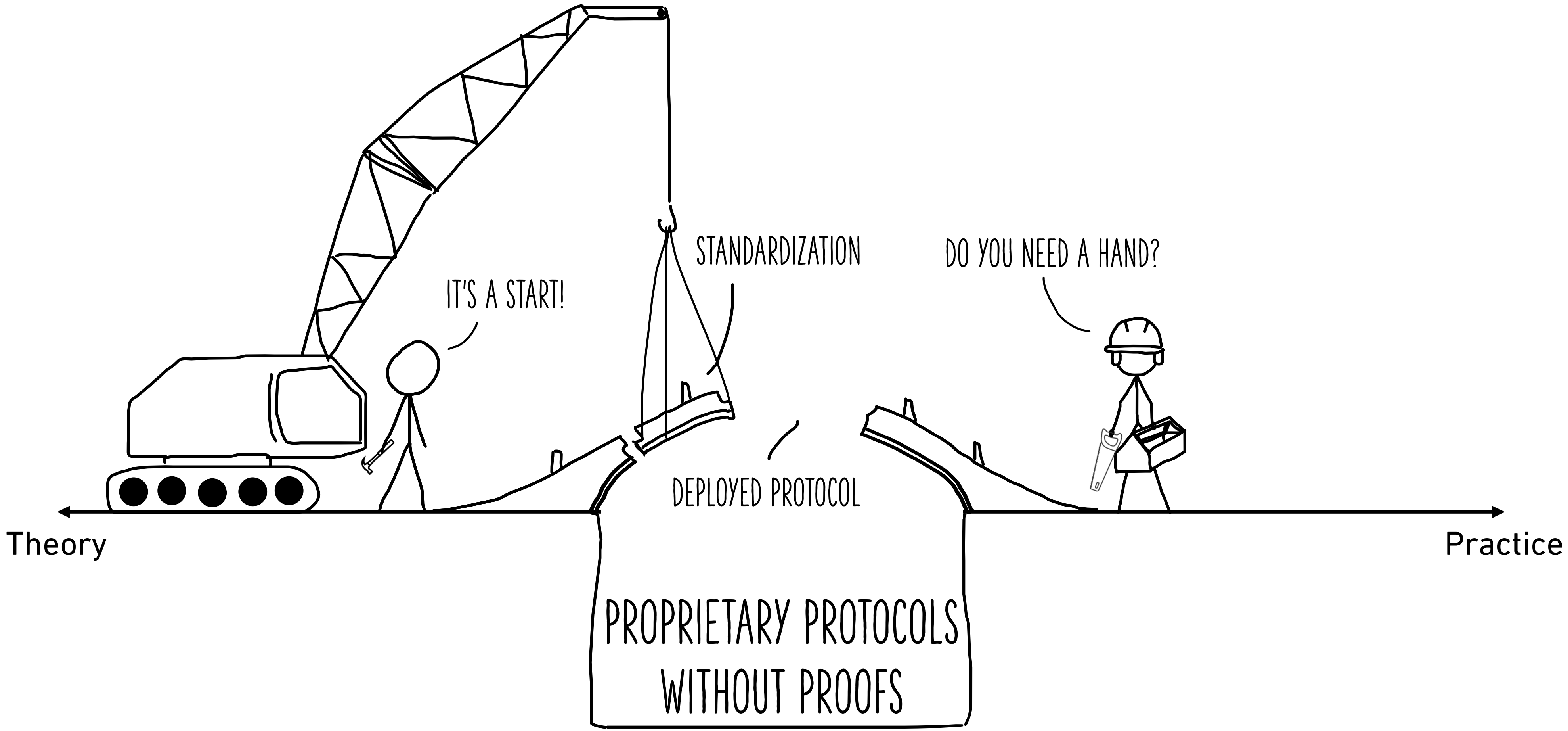
Security notions for E2EE cloud storage: game

Security game intuition

- Malicious server (adversary)
- Provide two files f_0, f_1
- File f_b is uploaded
- Guess bit $b' = b$
- Full control over state
- Users with correlated pws
- Oracles to make honest users perform actions
- User compromise



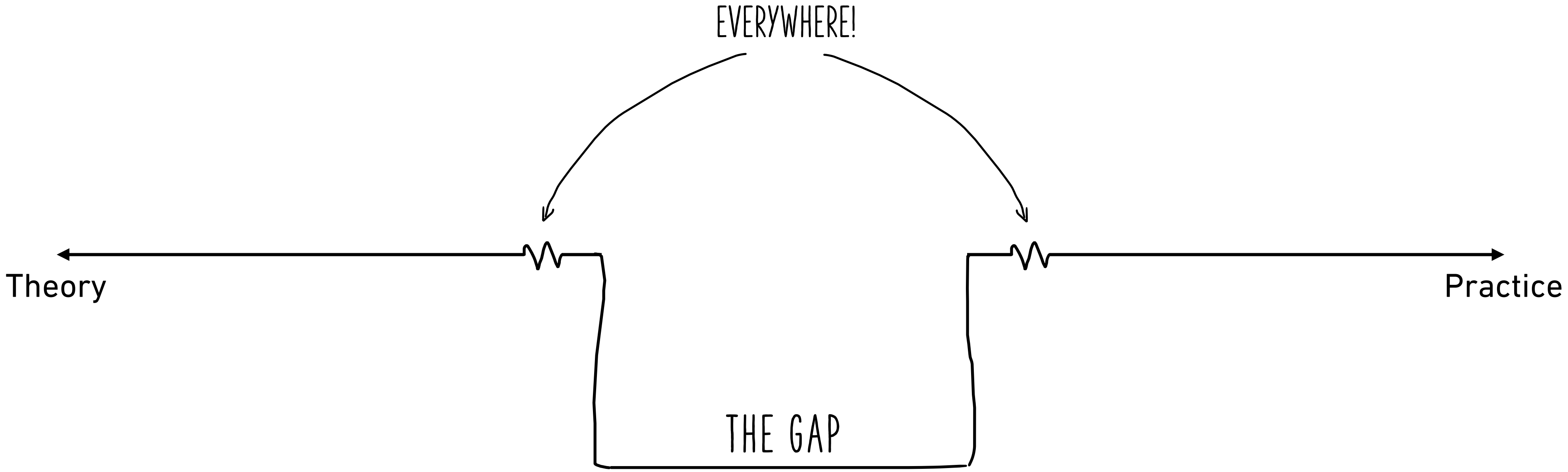
Building a Standard for E2EE Cloud Storage?



Conclusion

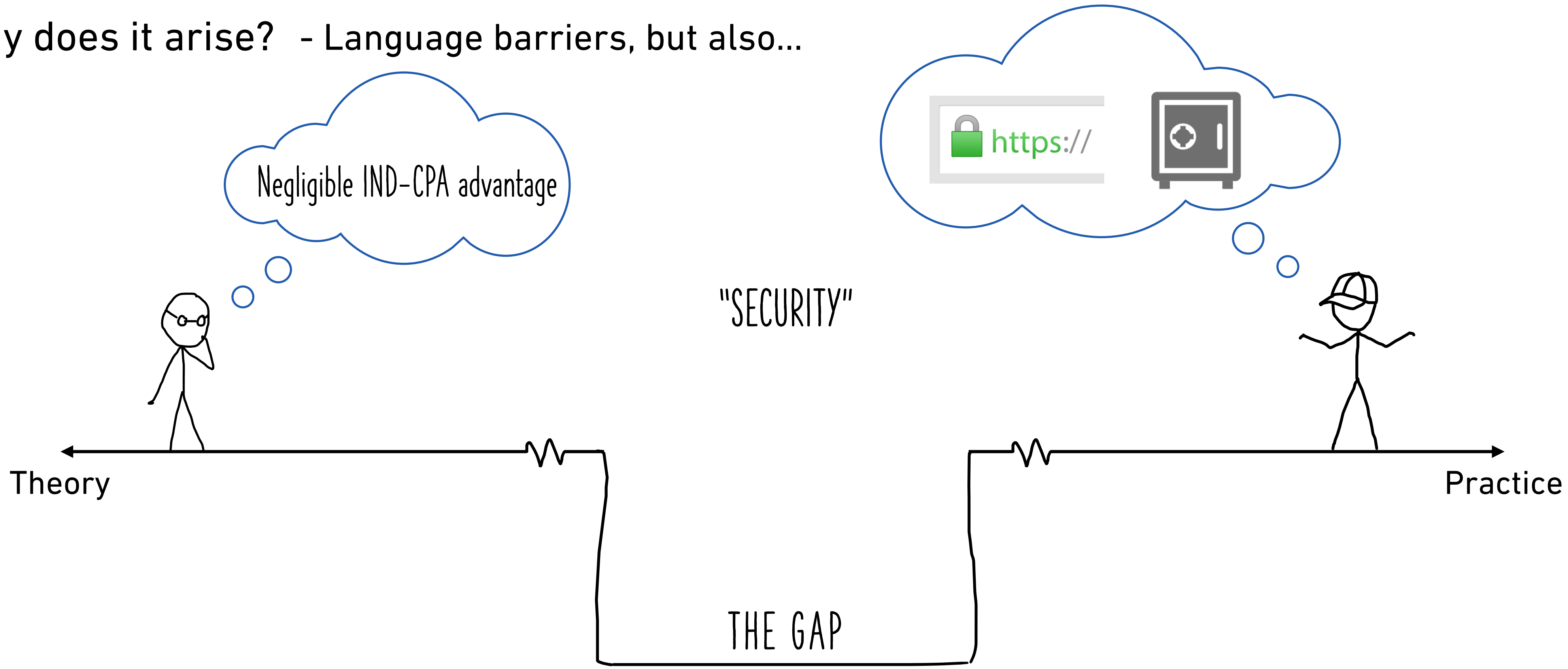
Conclusion

- Where does the gap arise?



Conclusion

- Where does the gap arise? - Everywhere
- Why does it arise? - Language barriers, but also…



Overstatements

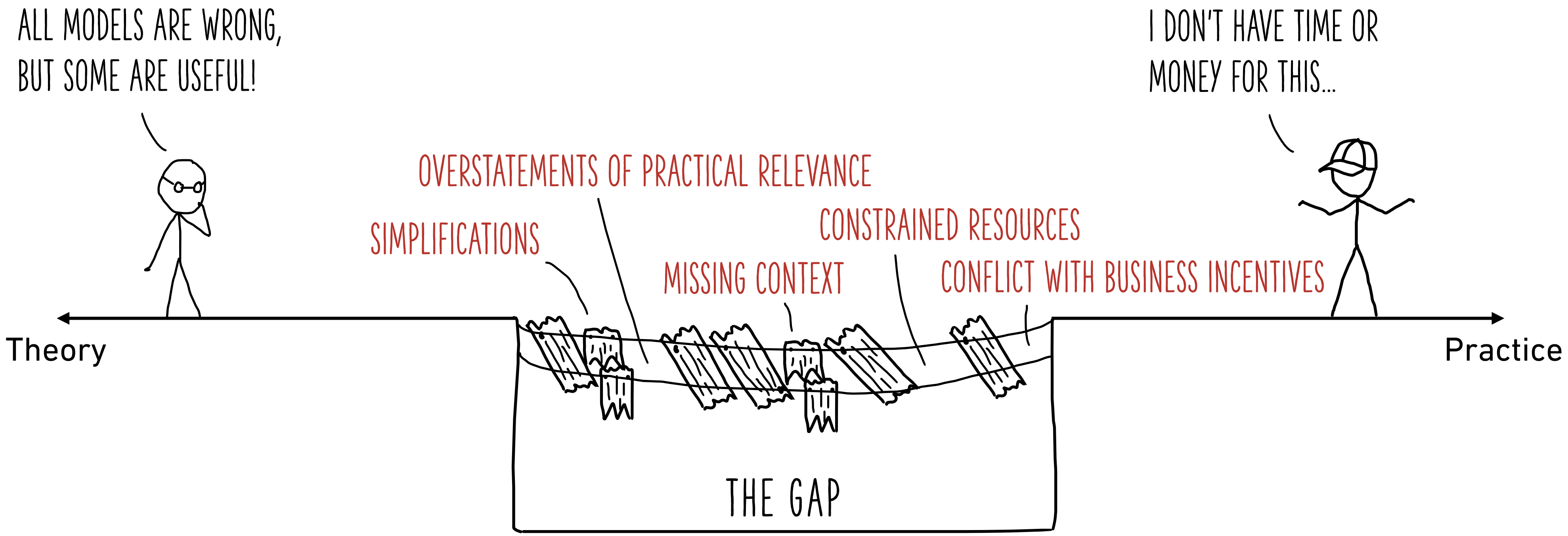
WHAT PEOPLE CLAIM THEY BUILT

WHAT THEY ACTUALLY BUILT

Image credit: Umer Sayyam, unsplash.com

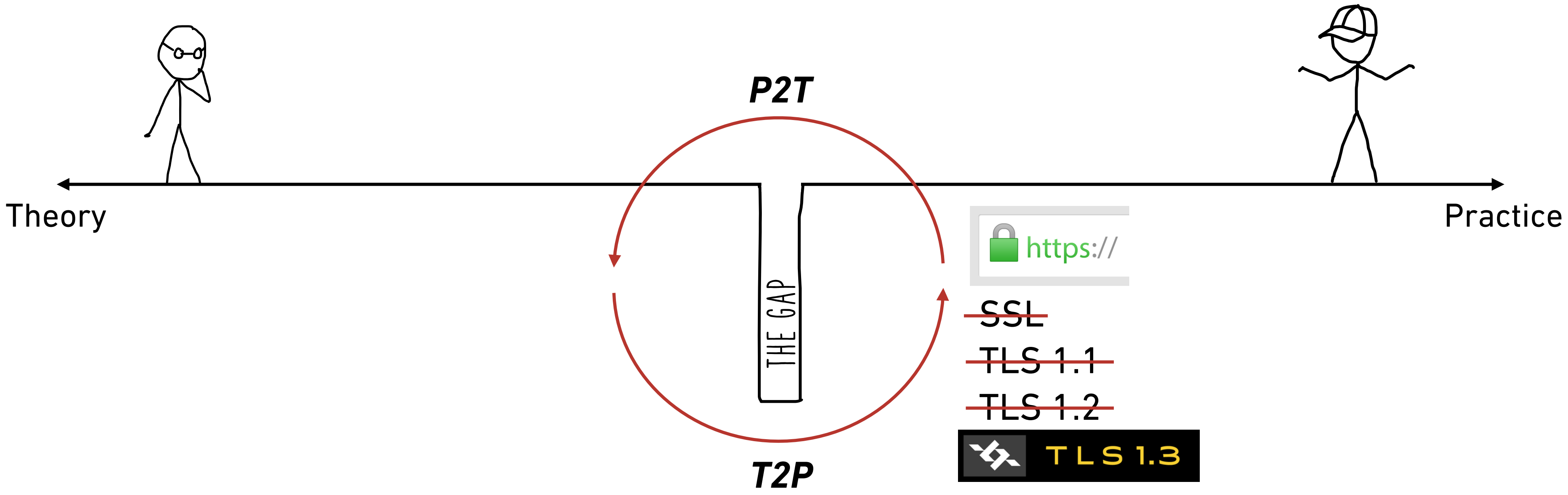
Conclusion

- Where does the gap arise? - Everywhere
- Why does it arise?

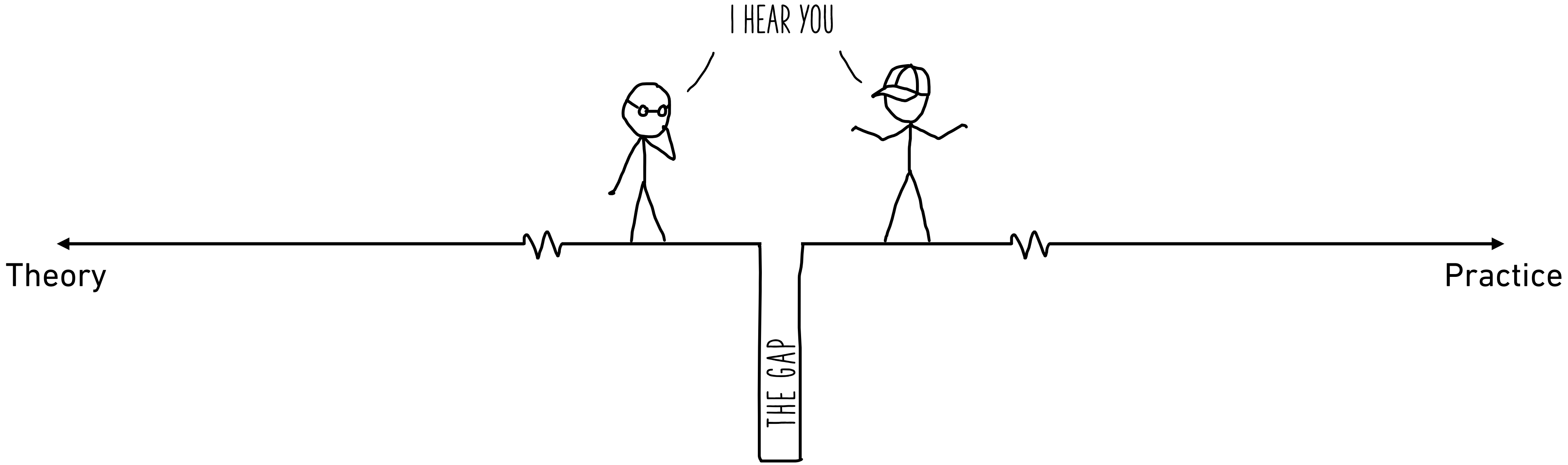


Conclusion

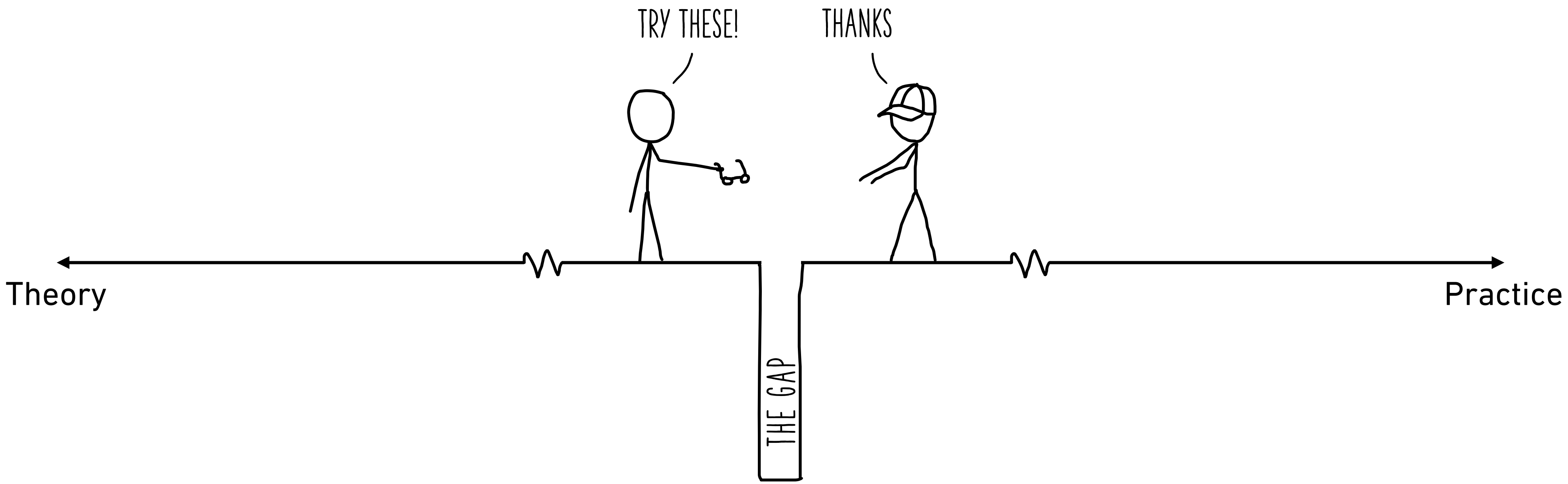
- Where does the gap arise? - Everywhere
- Why does it arise? - It's complicated
- Why is one loop of the cycle not enough to close the gap?



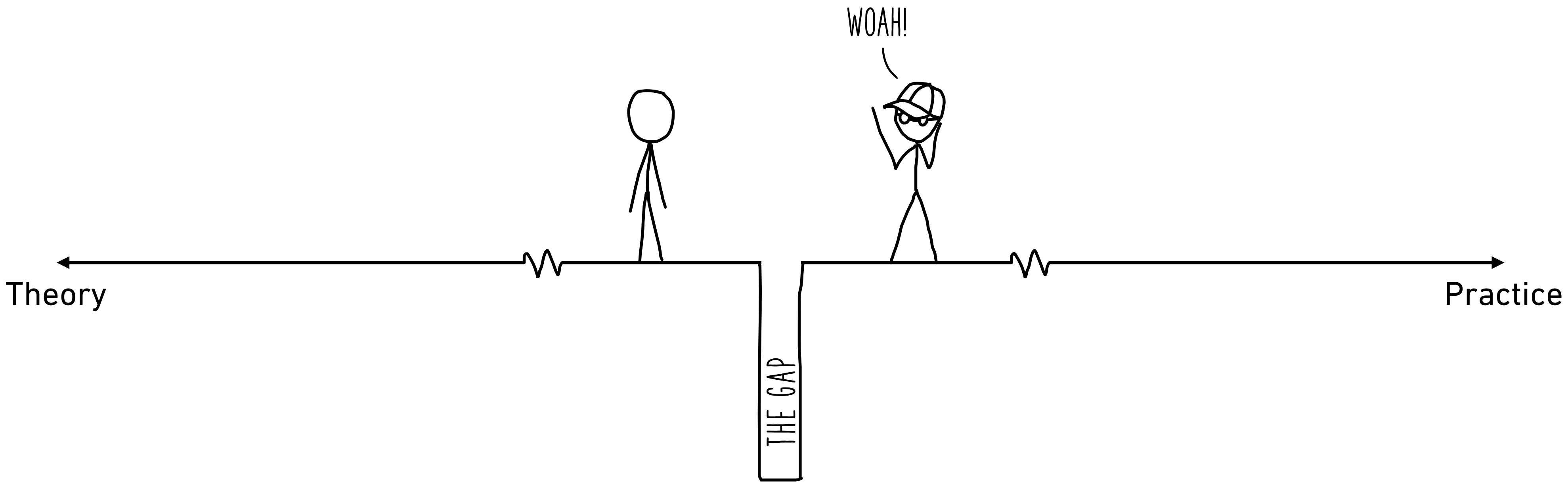
What Can We Do?



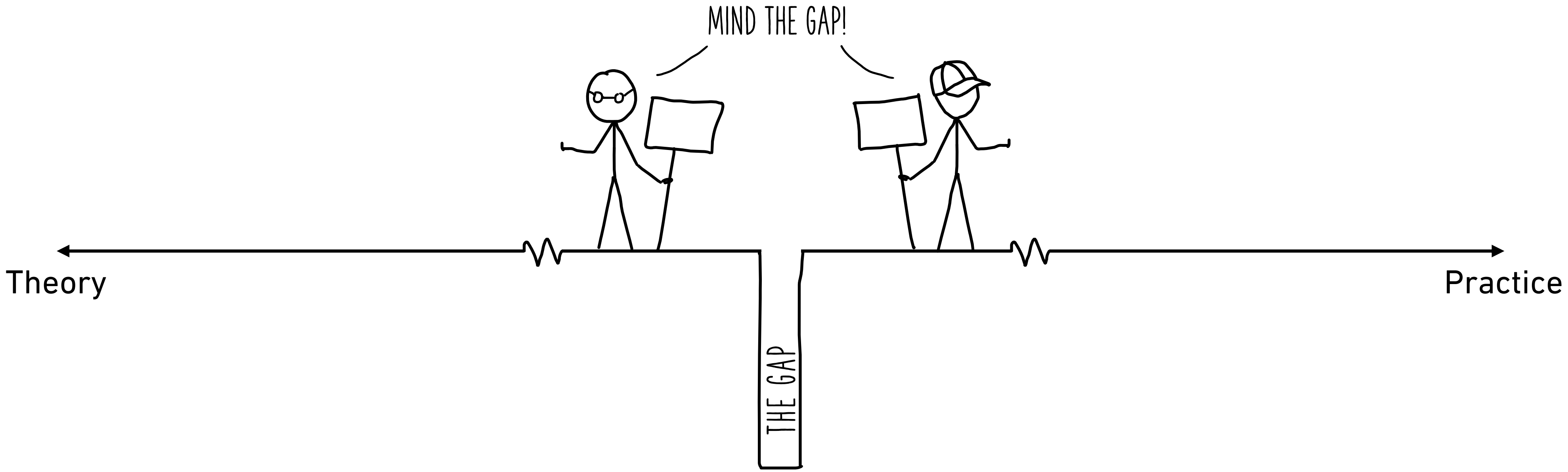
What Can We Do?



What Can We Do?



What Can We Do?



Why Should You Do Applied Cryptography?

- It’s impactful!
- It’s profitable!
- It’s fun!