On the Concrete Security of TLS 1.3 PSK Mode

Hannah Davis[1] **Denis Diemert**[2] Felix Günther[3] Tibor Jager[2]

[1]Department of Computer Science and Engineering
University of California San Diego

[2]IT Security and Cryptography Group
University of Wuppertal

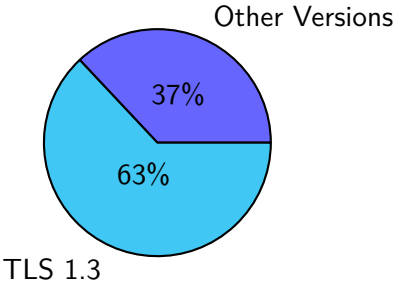
[3]Applied Cryptography Group
ETH Zurich

EUROCRYPT 2022

Transport Layer Security (TLS) 1.3

"TLS allows client/server applications to communicate over the Internet in a way that is designed to prevent eavesdropping, tampering, and message forgery."

— Abstract, RFC 8446

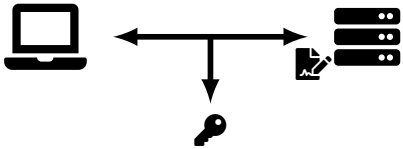


Preferred Version of the Top 1M Servers

— F5 Labs, Oct'21

Transport Layer Security (TLS) 1.3

Handshake Protocol

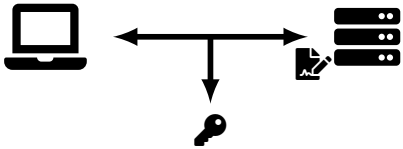


Application Data

Record Layer

Transport Layer Security (TLS) 1.3

Handshake Protocol

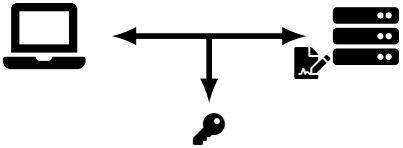


Record Layer

Application Data

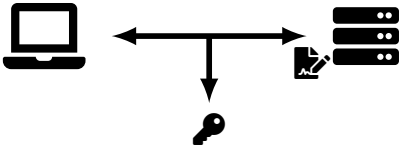
TLS 1.3 Handshake Protocol

TLS 1.3 Full Handshake

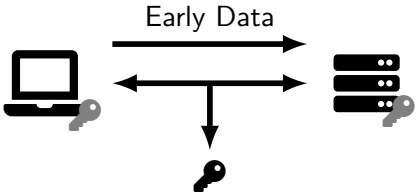


TLS 1.3 Handshake Protocol

TLS 1.3 Full Handshake

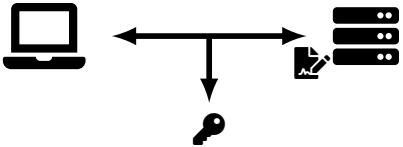


TLS 1.3 PSK-only Handshake

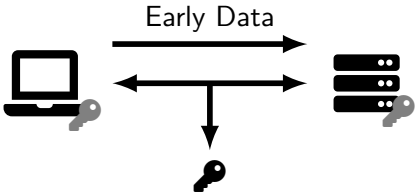


TLS 1.3 Handshake Protocol

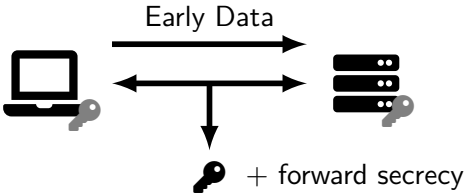
TLS 1.3 Full Handshake



TLS 1.3 PSK-only Handshake

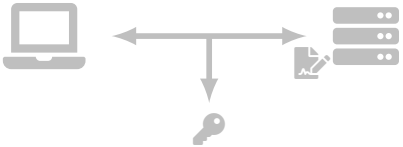


TLS 1.3 PSK-(EC)DHE Handshake

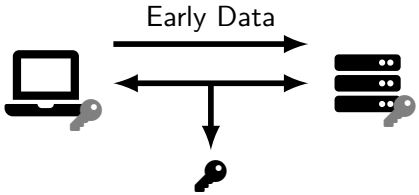


TLS 1.3 Handshake Protocol

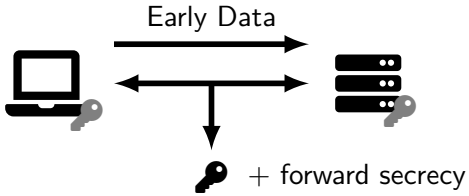
TLS 1.3 Full Handshake



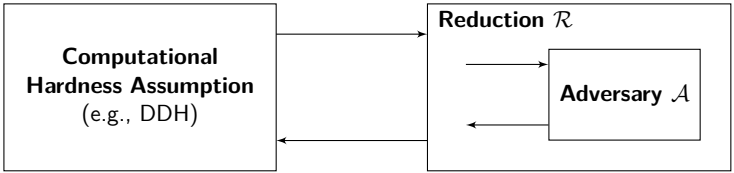
TLS 1.3 PSK-only Handshake



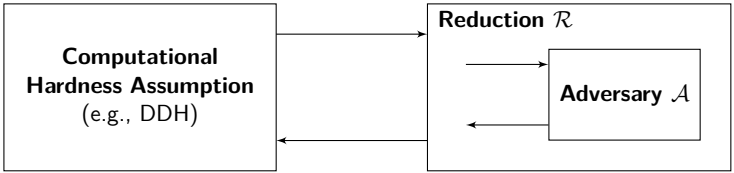
TLS 1.3 PSK-(EC)DHE Handshake



Concrete Security and Tightness



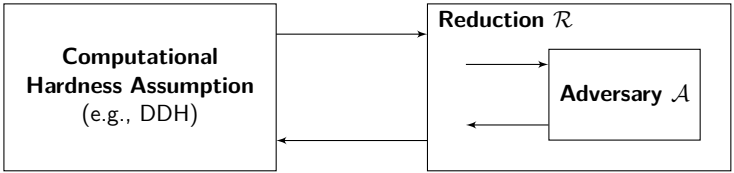
Concrete Security and Tightness



- Classically, security proofs considered asymptotically
  → Guarantee for the existence of "sufficiently" large parameters
- But: *no* guarantee that a real-world cryptosystem with standardized parameters achieves a certain expected security level
- Concrete security approach makes bounds explicit:

$$\mathsf{Adv}(\mathcal{A}) \leq \ell(\mathcal{A}) \cdot \mathsf{Adv}(\mathcal{R})$$

Concrete Security and Tightness



- Classically, security proofs considered asymptotically
  → Guarantee for the existence of "sufficiently" large parameters
- But: *no* guarantee that a real-world cryptosystem with standardized parameters achieves a certain expected security level
- Concrete security approach makes bounds explicit:

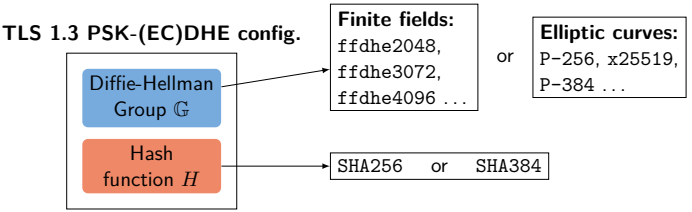
$$\mathsf{Adv}(\mathcal{A}) \leq \ell(\mathcal{A}) \cdot \mathsf{Adv}(\mathcal{R})$$

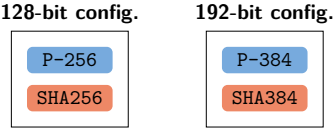
→ Tight security proof if ℓ is a small constant independent of $\mathcal{A}$

Previous Analyses of TLS 1.3

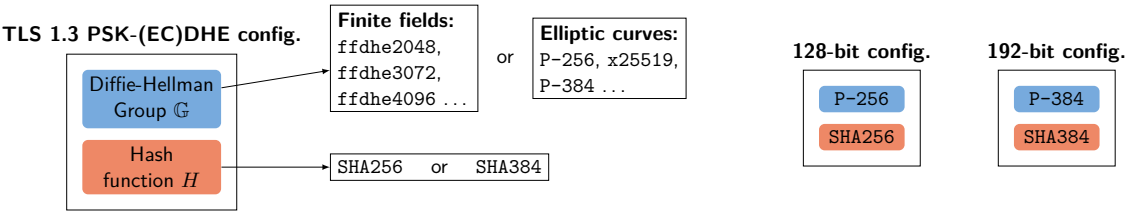
- TLS 1.3 developed in close collaboration between academia and industry
- Many draft revisions & final TLS 1.3 standard RFC 8446 were analyzed
- Dowling, Fischlin, Günther, and Stebila [DFGS21] gave bounds for
  - TLS 1.3 full
  - TLS 1.3 PSK-only
  - TLS 1.3 PSK-(EC)DHE

Security Bound for TLS 1.3 PSK-(EC)DHE by [DFGS21]





Security Bound for TLS 1.3 PSK-(EC)DHE by [DFGS21]

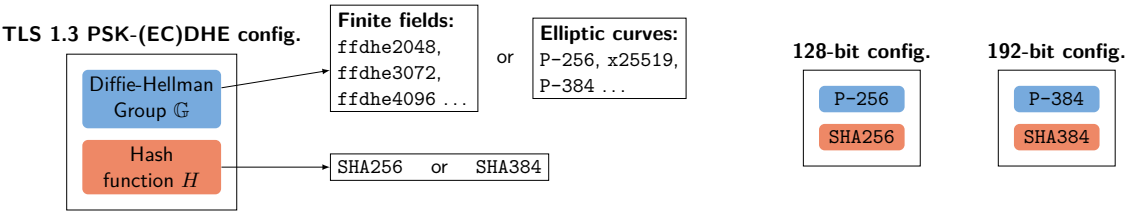


Theorem ([DFGS21], very simplified)

$$\mathsf{Adv}(\mathcal{A}, \text{TLS-PSK-(EC)DHE}) \leq 8S^2 \cdot \mathsf{Adv}(\mathcal{B}_1, \mathbb{G}) + 8S \cdot \mathsf{Adv}(\mathcal{B}_2, H) + \cdots$$

$\#$ sessions

Security Bound for TLS 1.3 PSK-(EC)DHE by [DFGS21]



Highly non-tight!

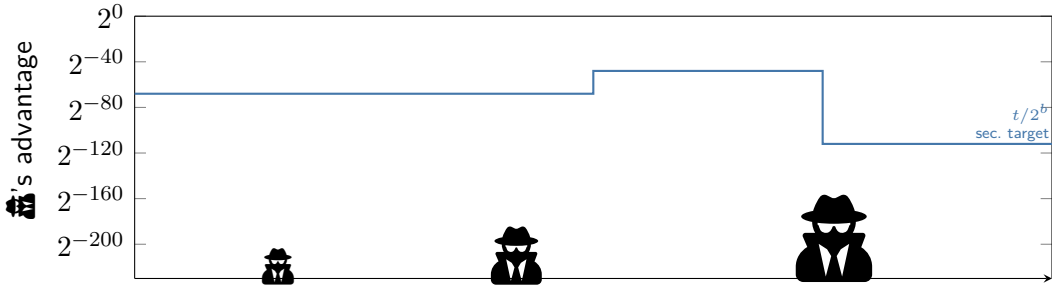
Theorem ([DFGS21], very simplified)

$$\mathsf{Adv}(\mathcal{A}, \text{TLS-PSK-(EC)DHE}) \leq 8S^2 \cdot \mathsf{Adv}(\mathcal{B}_1, \mathbb{G}) + 8S \cdot \mathsf{Adv}(\mathcal{B}_2, H) + \cdots$$

$\#$ sessions

Concrete Evaluation of [DFGS21]'s Bound

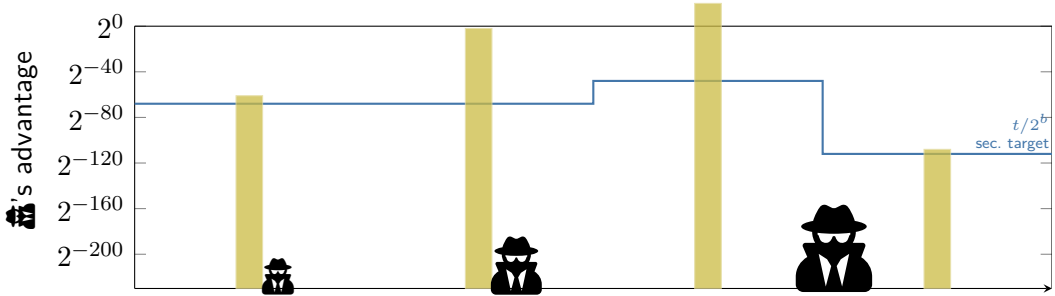
- **[DFGS21]**: $\mathsf{Adv}(\text{🕵}, \text{TLS-PSK-(EC)DHE}) \leq 8S^2 \cdot \mathsf{Adv}(\mathcal{B}_1, \mathbb{G}) + 8S \cdot \mathsf{Adv}(\mathcal{B}_2, H) + \cdots$



t: attacker time
$\#N$: number of PSKs
$\#S$: number of sessions

Concrete Evaluation of [DFGS21]'s Bound

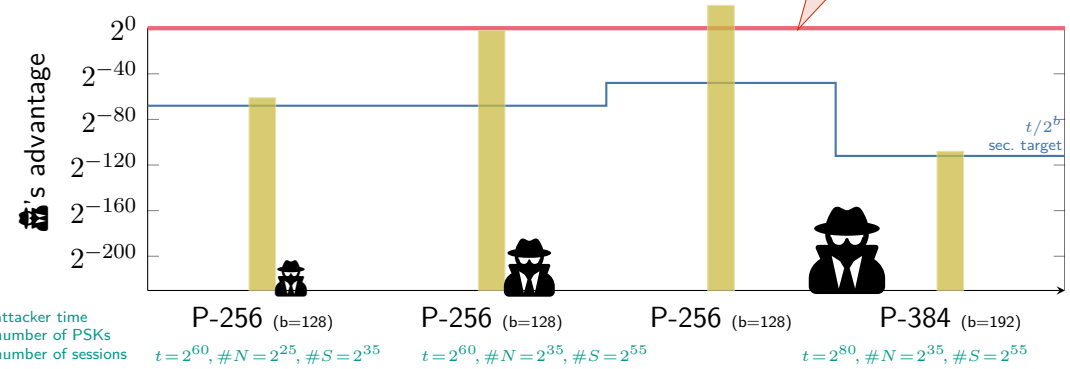
- **[DFGS21]**: $\mathsf{Adv}(\,\text{🕵}\,, \text{TLS-PSK-(EC)DHE}) \leq 8S^2 \cdot \mathsf{Adv}(\mathcal{B}_1, \mathbb{G}) + 8S \cdot \mathsf{Adv}(\mathcal{B}_2, H) + \cdots$



t: attacker time
$\#N$: number of PSKs
$\#S$: number of sessions

Concrete Evaluation of [DFGS21]'s Bound

- **[DFGS21]**: $\mathsf{Adv}(\,\text{🕵}\,, \text{TLS-PSK-(EC)DHE}) \leq 8S^2 \cdot \mathsf{Adv}(\mathcal{B}_1, \mathbb{G}) + \cdots + 8S \cdot \mathsf{Adv}(\mathcal{B}_2, H) + \cdots$



t: attacker time
$\#N$: number of PSKs
$\#S$: number of sessions

Tighter Bounds for TLS 1.3

Tighter Bounds for TLS 1.3 Full Handshake

- Davis and Günther [DG21] and Diemert and Jager [DJ21] independently gave tighter bounds for full handshake
- Limitations:
  - Both made assumptions about the key schedule (next)
  - Signatures need to be multi-user-secure with adaptive corruptions
    → none of the standardized signatures satisfies this tightly (implicit linear loss)

Tighter Bounds for TLS 1.3

Tighter Bounds for TLS 1.3 Full Handshake

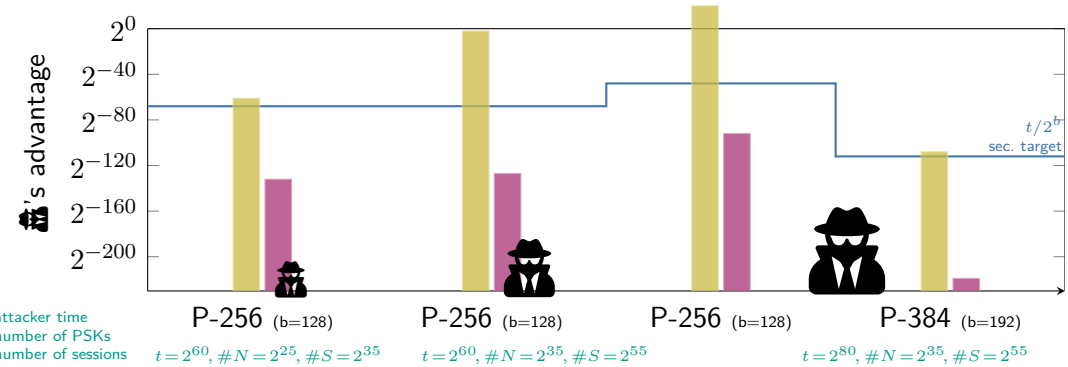
- Davis and Günther [DG21] and Diemert and Jager [DJ21] independently gave tighter bounds for full handshake
- Limitations:
  - Both made assumptions about the key schedule (next)
  - Signatures need to be multi-user-secure with adaptive corruptions
    → none of the standardized signatures satisfies this tightly (implicit linear loss)

Tight Bounds for TLS 1.3 PSK mode

- In this work, we give fully tight bounds for
  - TLS 1.3 PSK-only and TLS 1.3 PSK-(EC)DHE
- Exception: PSK-only with `SHA384`

Tight Bounds for TLS 1.3 PSK-(EC)DHE

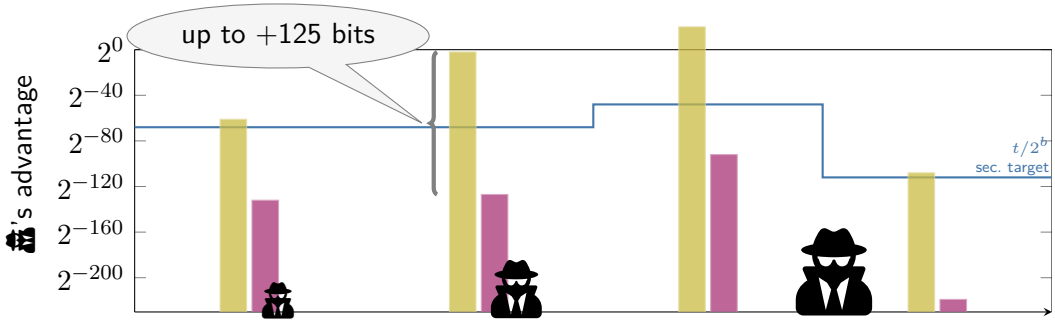
- **[DFGS21]**: $\mathsf{Adv}(\mathcal{A}, \text{TLS-PSK-(EC)DHE}) \leq 8S^2 \cdot \mathsf{Adv}(\mathcal{B}_1, \mathbb{G}) + 8S \cdot \mathsf{Adv}(\mathcal{B}_2, H) + \cdots$
- **Us**: $\mathsf{Adv}(\mathcal{A}, \text{TLS-PSK-(EC)DHE}) \leq \mathsf{Adv}(\mathcal{B}_1, \mathbb{G}) + \mathsf{Adv}(\mathcal{B}_2, H) + \cdots$



t: attacker time
$\#N$: number of PSKs
$\#S$: number of sessions

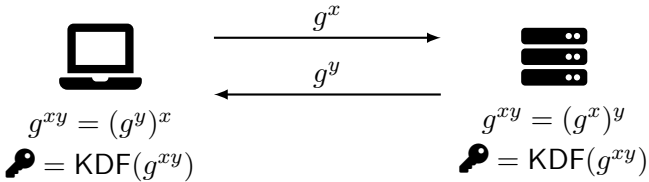
Tight Bounds for TLS 1.3 PSK-(EC)DHE

- **[DFGS21]**: $\mathsf{Adv}(\mathcal{A}, \text{TLS-PSK-(EC)DHE}) \leq 8S^2 \cdot \mathsf{Adv}(\mathcal{B}_1, \mathbb{G}) + 8S \cdot \mathsf{Adv}(\mathcal{B}_2, H) + \cdots$
- **Us**: $\mathsf{Adv}(\mathcal{A}, \text{TLS-PSK-(EC)DHE}) \leq \mathsf{Adv}(\mathcal{B}_1, \mathbb{G}) + \mathsf{Adv}(\mathcal{B}_2, H) + \cdots$



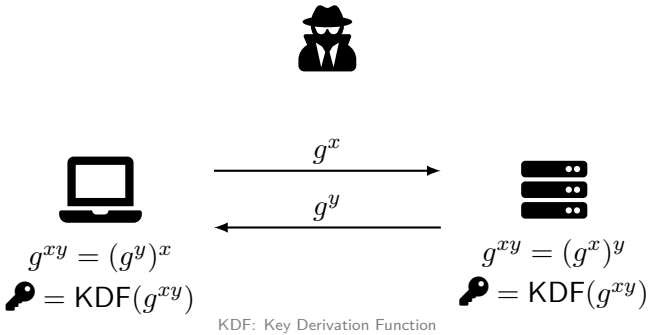
t: attacker time
$\#N$: number of PSKs
$\#S$: number of sessions

Why are the results not inherently tight?



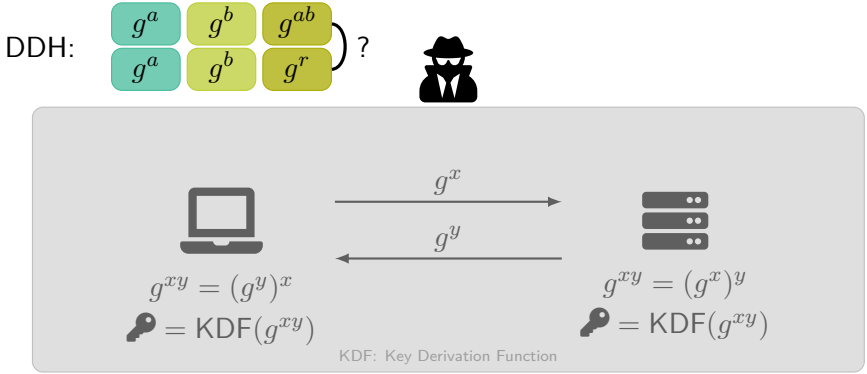
KDF: Key Derivation Function

Why are the results not inherently tight?



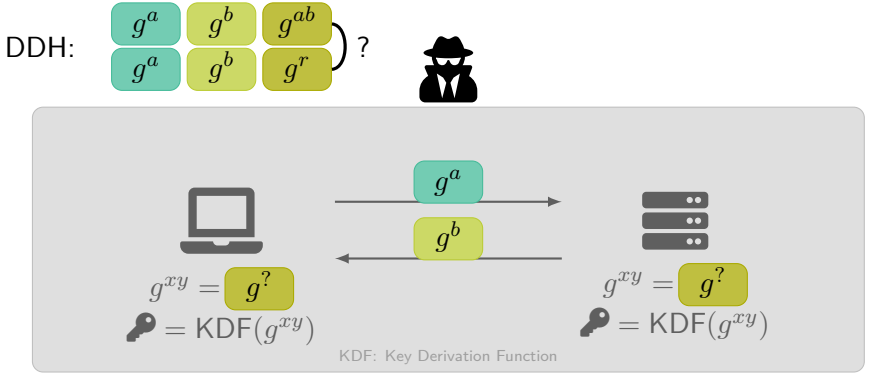
- key secrecy: given only g^x and g^y, key 🔑 remains secret (indist. from 🎲)

Why are the results not inherently tight?



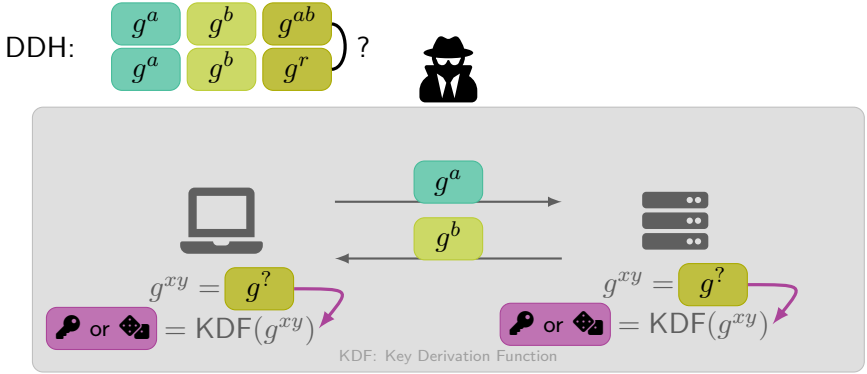
- key secrecy: given only g^x and g^y, key 🔑 remains secret (indist. from 🎲)

Why are the results not inherently tight?



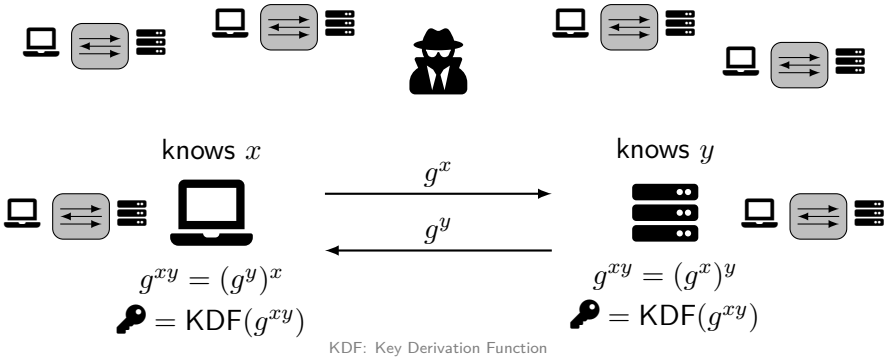
- key secrecy: given only g^x and g^y, key 🔑 remains secret (indist. from 🎲)

Why are the results not inherently tight?



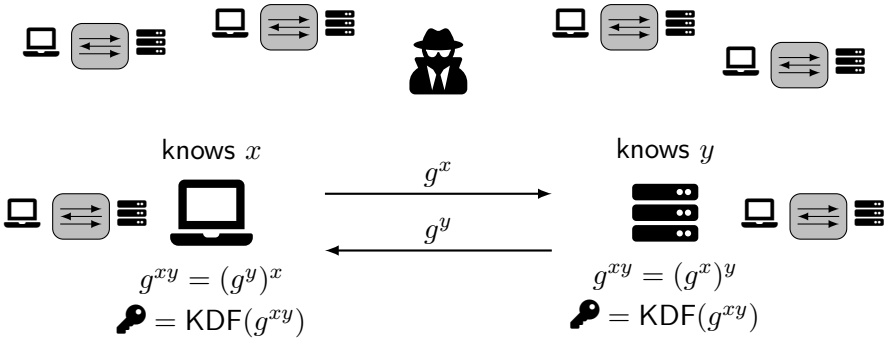
- key secrecy: given only g^x and g^y, key 🔑 remains secret (indist. from 🎲)

Why are the results not inherently tight?



- key secrecy: given only g^x and g^y, key 🔑 remains secret (indist. from 🎲)
- Problem: many sessions in parallel... Where to embed?

Why are the results not inherently tight?



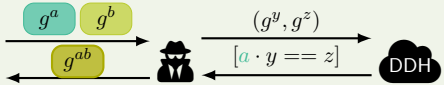
- key secrecy: given only g^x and g^y, key 🔑 remains secret (indist. from 🎲)
- Problem: many sessions in parallel... Where to embed?
  - Solution: guess 💻 and 🖴 $\implies$ loss of $(\#\text{ session})^2$

Tighter Proof for "simple DH" by [Coh+19]

Reduction idea

1. Embed rerandomized g^a in all and rerandomized g^b in all
2. The key derivation of must be modeled as random oracle RO.
   - Crucial: KDF gets as input $(g^x, g^y, g^{xy}, ...)$
   - Detect "correct" RO queries with DDH oracle $\rightsquigarrow$ solving StrongDH (instead of DDH)

StrongDH ("Solve CDH with DDH oracle"):

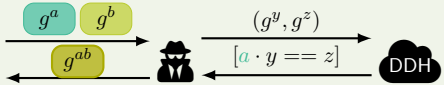


Tighter Proof for "simple DH" by [Coh+19]

Reduction idea

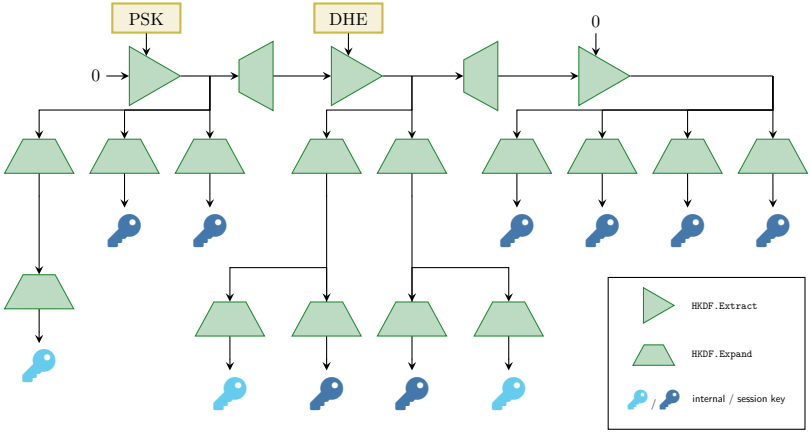
1. Embed rerandomized g^a in all 💻 and rerandomized g^b in all 🖴
2. The key derivation of 🔑 must be modeled as random oracle RO.
   - Crucial: KDF gets as input $(g^x, g^y, g^{xy}, ...)$
   - Detect "correct" RO queries with DDH oracle $\rightsquigarrow$ solving StrongDH (instead of DDH)

StrongDH ("Solve CDH with DDH oracle"):

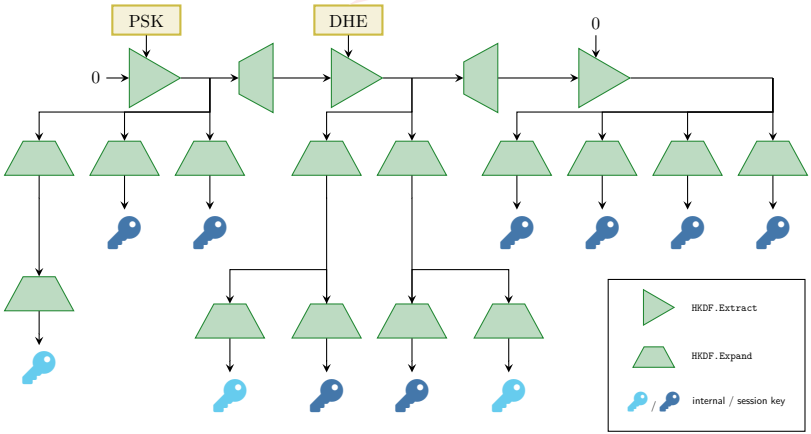


Is this reduction idea a template for a tight(er) proof for TLS 1.3?

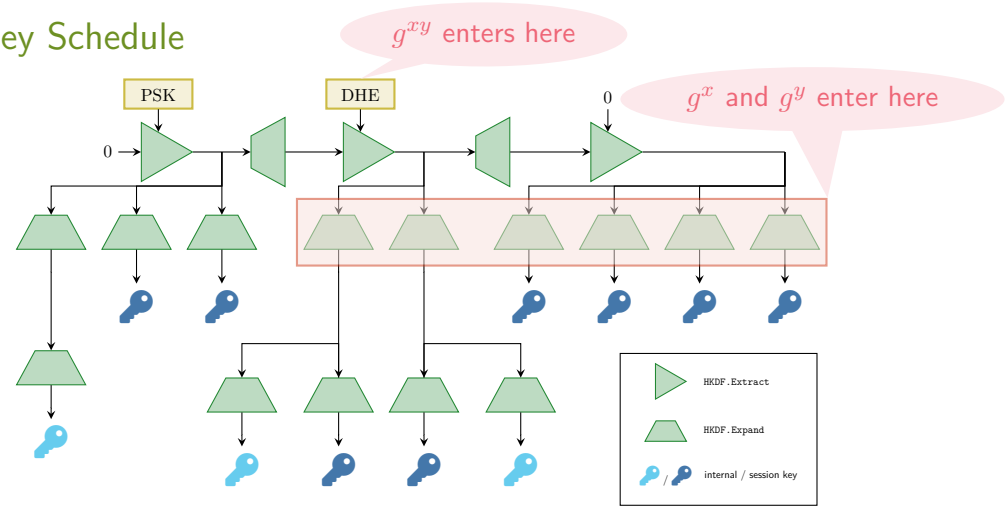
TLS 1.3 Key Schedule



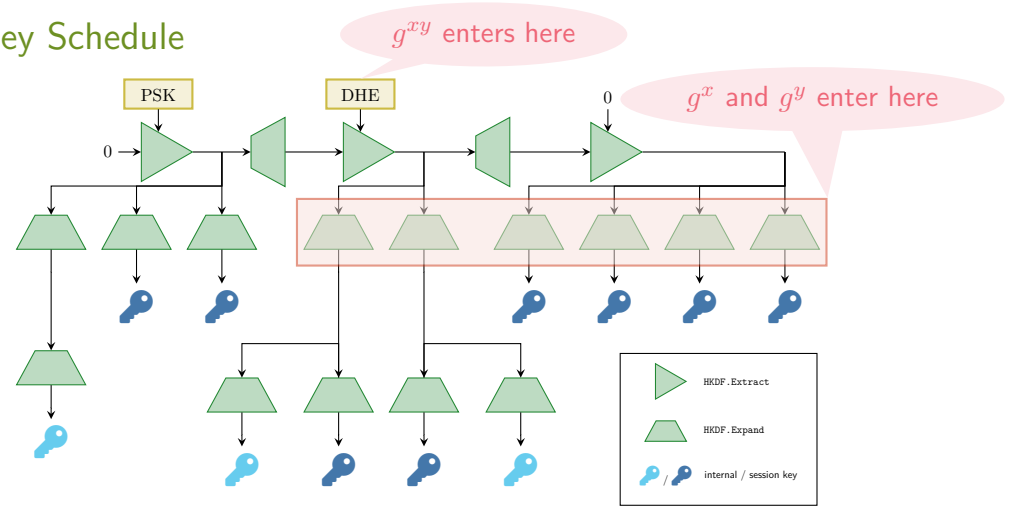
TLS 1.3 Key Schedule



TLS 1.3 Key Schedule

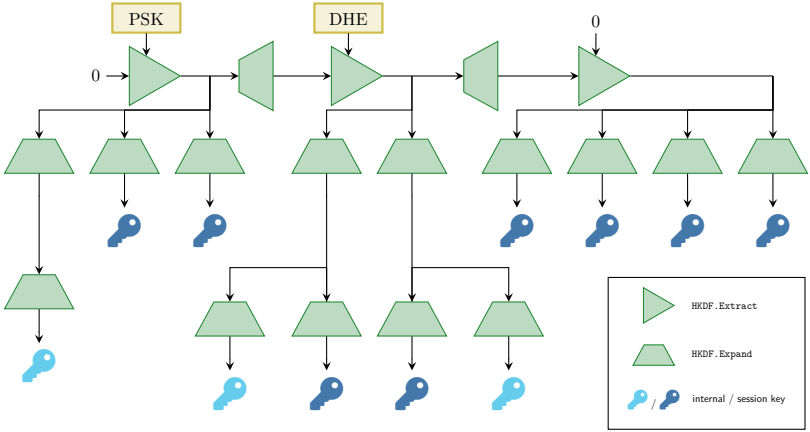


TLS 1.3 Key Schedule

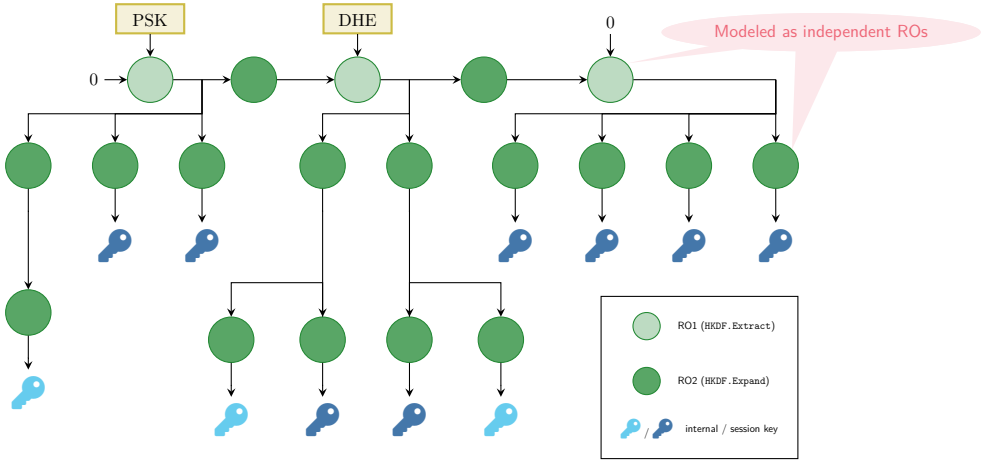


[Coh+19] technique not directly applicable!

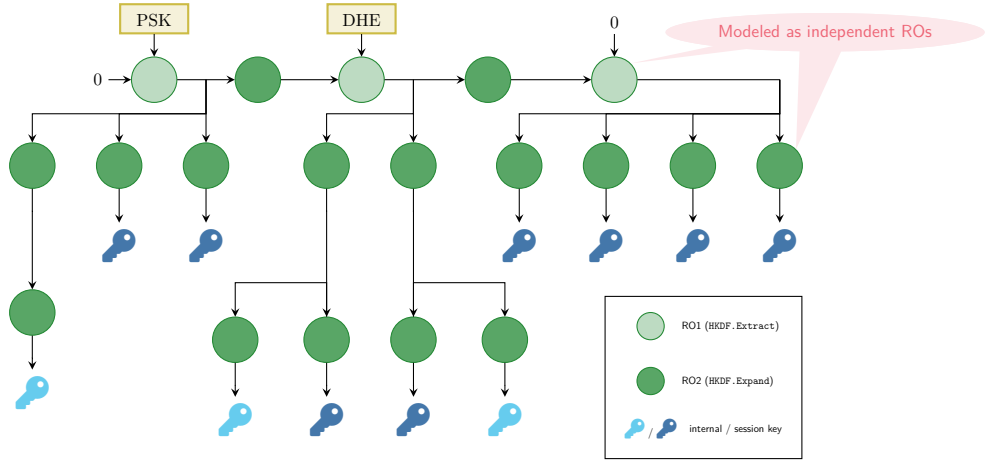
Solution 1 – [DG21]



Solution 1 – [DG21]



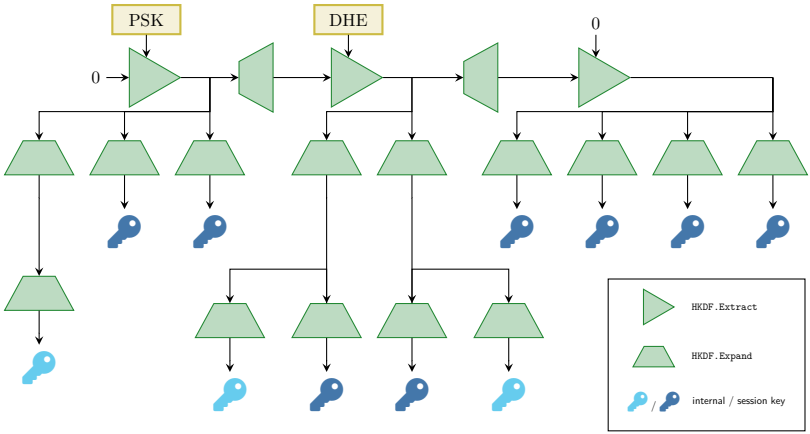
Solution 1 – [DG21]



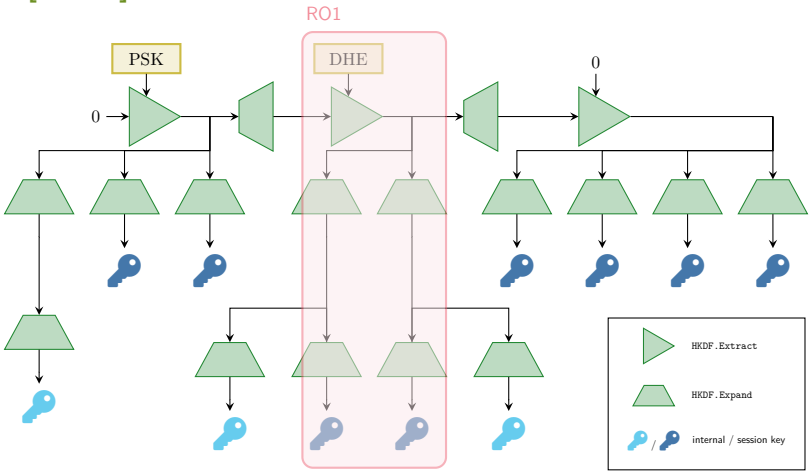
HKDF.Extract and HKDF.Expand unfortunately are not independent!

$\texttt{HKDF.Extract}(K, M) = \text{HMAC}(K, M)$ $\texttt{HKDF.Expand}(\text{K,M}) = \text{HMAC}(K, M \parallel \texttt{0x01})$

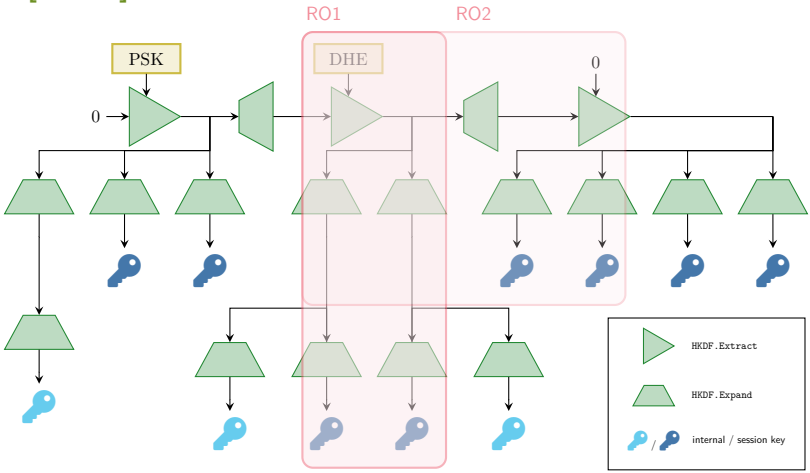
Solution 2 – [DJ21]



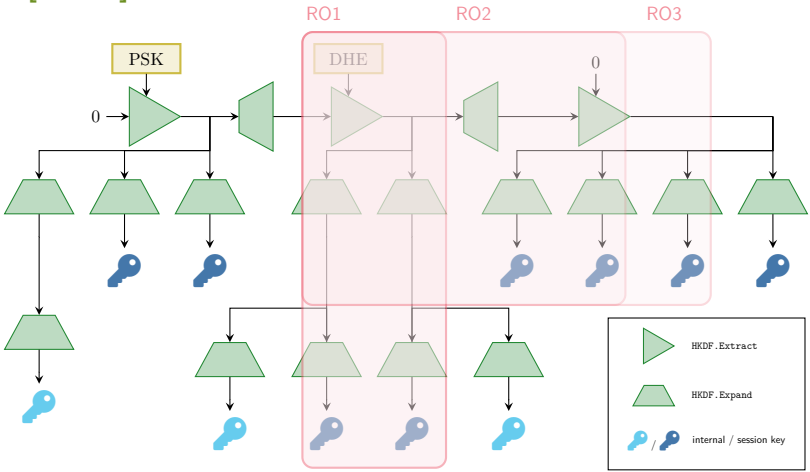
Solution 2 – [DJ21]



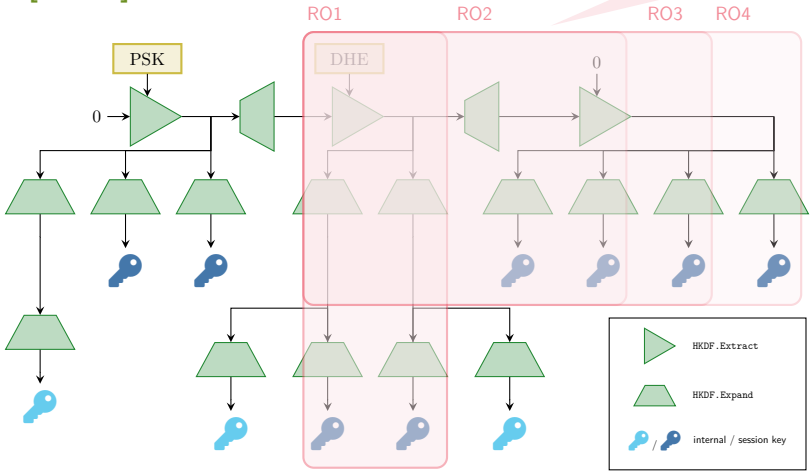
Solution 2 – [DJ21]



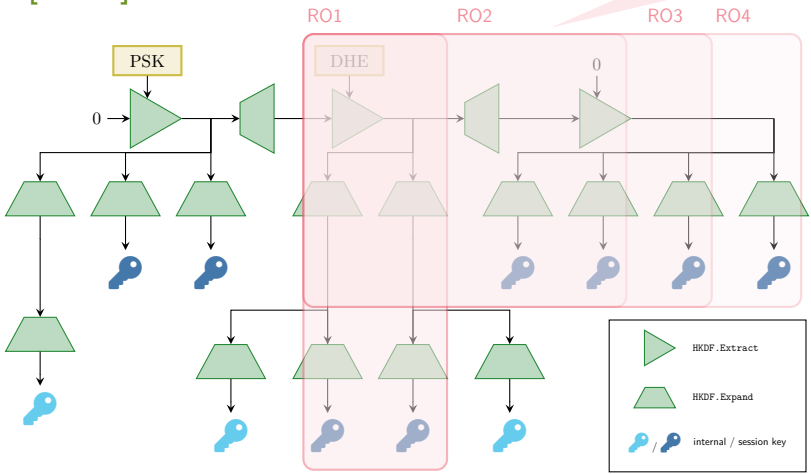
Solution 2 – [DJ21]



Solution 2 – [DJ21]

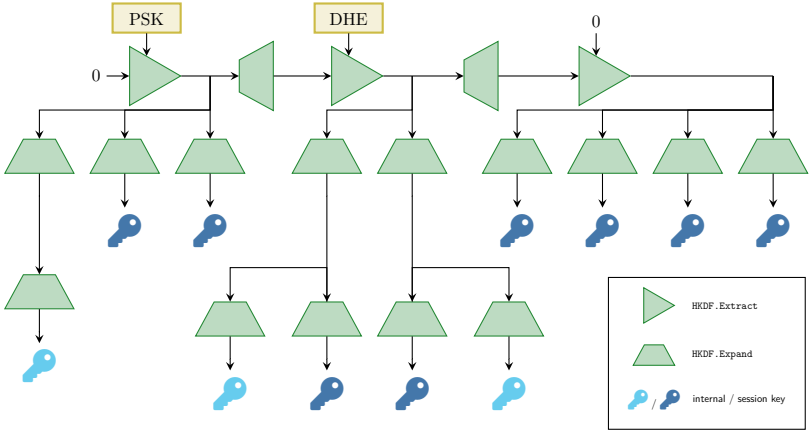


Solution 2 – [DJ21]

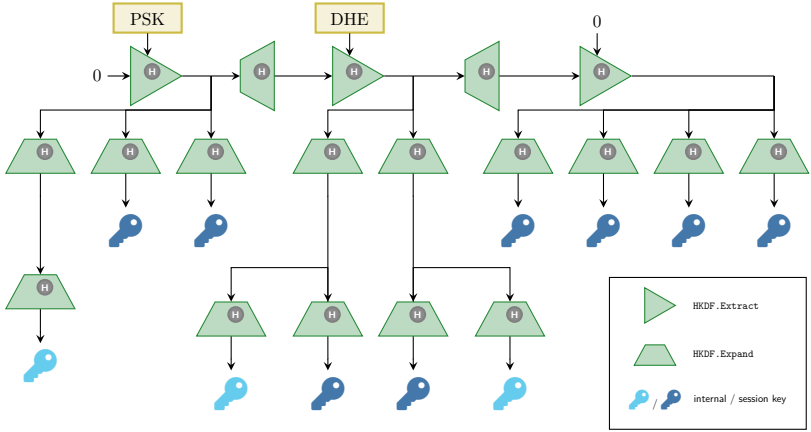


Proof becomes more direct, however assumption not justfied

Bottom Line of the Solutions of DG and DJ

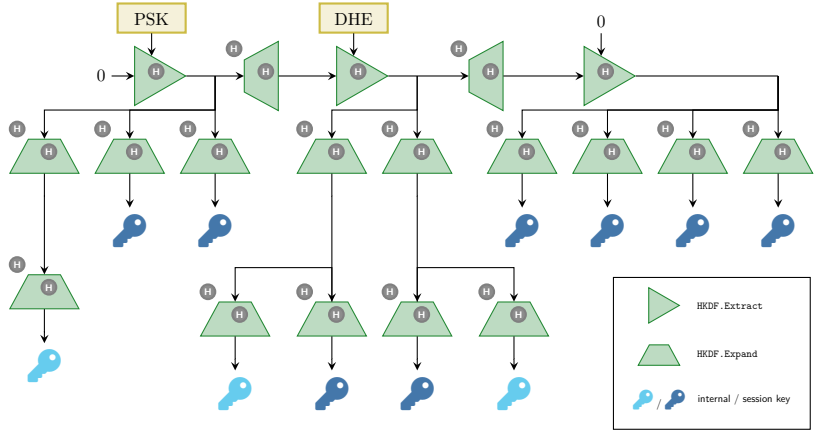


Bottom Line of the Solutions of DG and DJ



Both miss: TLS uses the same hash function for different purposes in key schedule!

Bottom Line of the Solutions of DG and DJ



Both miss: TLS uses the same hash function for different purposes in key schedule!

Our Modularization using Indifferentiability [MRH04]

Theorem

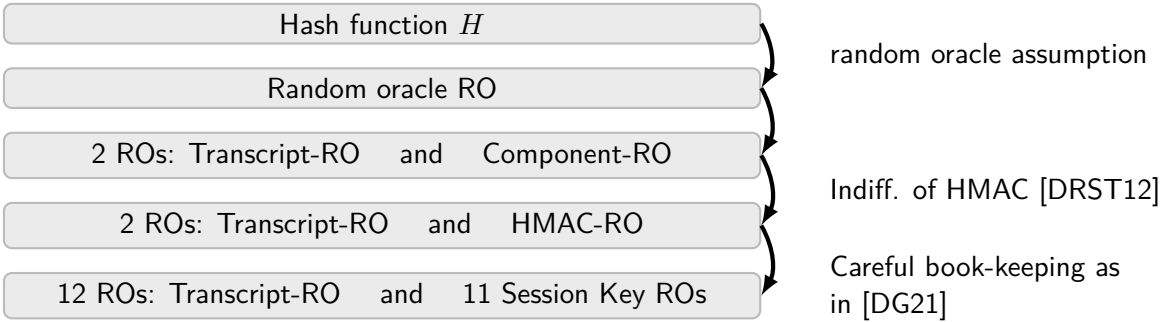
$$\mathsf{Adv}(\mathcal{A}, \text{TLS-PSK}) \leq \mathsf{Adv}(\mathcal{B}_1, \text{Abstracted TLS-PSK}) + \mathsf{Adv}^{\text{Indiff.}}(\mathcal{B}_2, \text{Abstraction})$$

Our Modularization using Indifferentiability [MRH04]

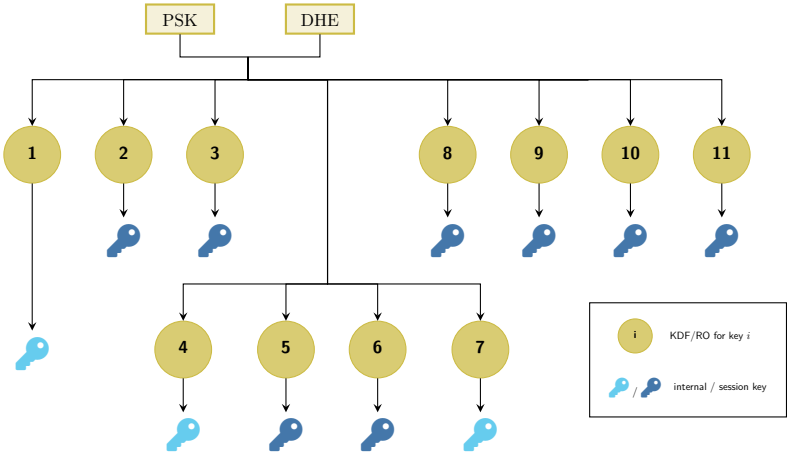
Theorem

$$\mathsf{Adv}(\mathcal{A}, \text{TLS-PSK}) \leq \mathsf{Adv}(\mathcal{B}_1, \text{Abstracted TLS-PSK}) + \mathsf{Adv}^{\text{Indiff.}}(\mathcal{B}_2, \text{Abstraction})$$

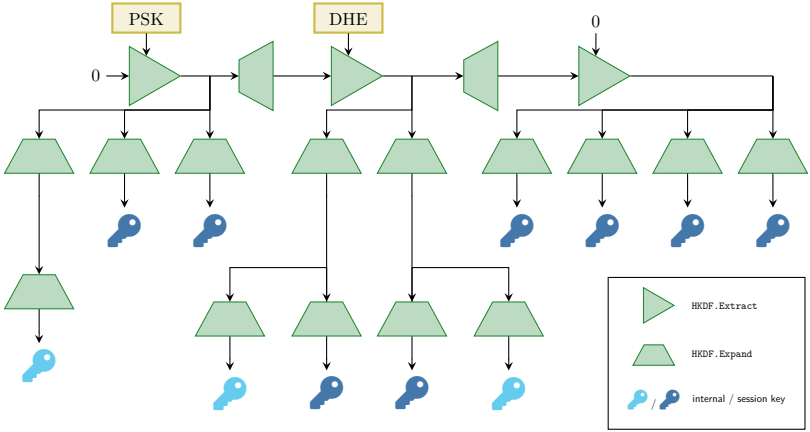
Justifying the Abstraction:



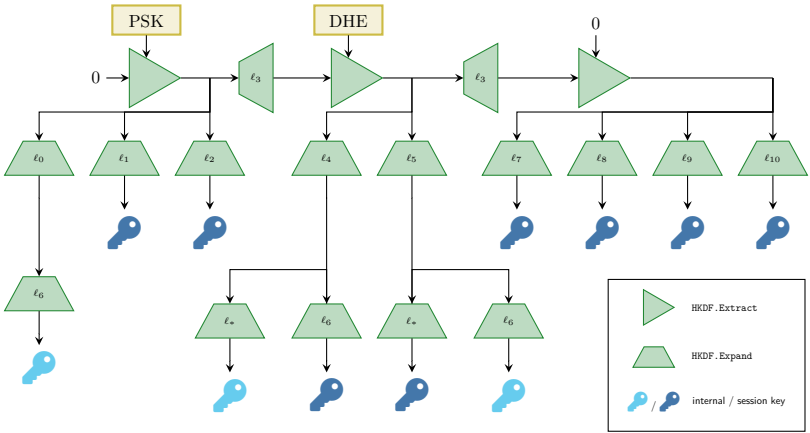
Our Key Schedule Abstraction (simplified)



HMAC-RO $\Longrightarrow$ 11 Session Key ROs

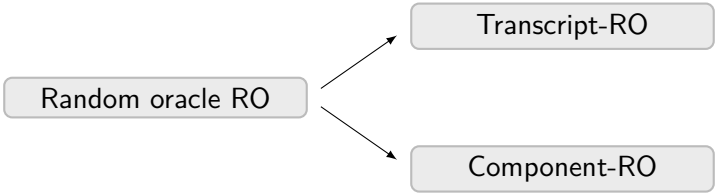


HMAC-RO $\Longrightarrow$ 11 Session Key ROs



➔ `HKDF.Expand` is explicitly domain separated using labels
$\texttt{HKDF.Expand}(K, \ell \,\|\, \cdots) = \mathsf{HMAC}(K, \ell \,\|\, \cdots \|\, \texttt{0x01})$

Lack of Domain Separation



- No explicit domain separation (e.g., labels) for uses of hash function
- Separating the uses of the hash function = distinguishing inputs

$$\text{input} \stackrel{?}{=} \text{transcript / HMAC call / HDKF call}$$

Domain separation works except for PSK-only with SHA384 mode

Summary

- Tight bounds for the TLS 1.3 PSK modes and a concrete evaluation
- New abstraction of the TLS 1.3 key schedule used in the PSK mode
- Identify a lack of domain separation in TLS 1.3 PSK-only with SHA384

Summary

- Tight bounds for the TLS 1.3 PSK modes and a concrete evaluation
- New abstraction of the TLS 1.3 key schedule used in the PSK mode
- Identify a lack of domain separation in TLS 1.3 PSK-only with SHA384

Thank you!

https://ia.cr/2022/246 diemert@uni-wuppertal.de

References I

[Coh+19] K. Cohn-Gordon, C. Cremers, K. Gjøsteen, H. Jacobsen, and T. Jager. "Highly Efficient Key Exchange Protocols with Optimal Tightness". In: *CRYPTO 2019, Part III*. Ed. by A. Boldyreva and D. Micciancio. Vol. 11694. LNCS. Springer, Heidelberg, Aug. 2019, pp. 767–797.

[DG21] H. Davis and F. Günther. "Tighter Proofs for the SIGMA and TLS 1.3 Key Exchange Protocols". In: *19th International Conference on Applied Cryptography and Network Security (ACNS 2021)*. 2021.

[DJ21] D. Diemert and T. Jager. "On the Tight Security of TLS 1.3: Theoretically Sound Cryptographic Parameters for Real-World Deployments". In: *Journal of Cryptology* 34.3 (July 2021), p. 30.

[DRST12] Y. Dodis, T. Ristenpart, J. P. Steinberger, and S. Tessaro. "To Hash or Not to Hash Again? (In)Differentiability Results for H^2 and HMAC". In: *CRYPTO 2012*. Ed. by R. Safavi-Naini and R. Canetti. Vol. 7417. LNCS. Springer, Heidelberg, Aug. 2012, pp. 348–366.

[DFGS21] B. Dowling, M. Fischlin, F. Günther, and D. Stebila. "A Cryptographic Analysis of the TLS 1.3 Handshake Protocol". In: *Journal of Cryptology* 34.4 (Oct. 2021), p. 37.

[MRH04] U. M. Maurer, R. Renner, and C. Holenstein. "Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology". In: *TCC 2004*. Ed. by M. Naor. Vol. 2951. LNCS. Springer, Heidelberg, Feb. 2004, pp. 21–39.