BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures





Cas Cremers[1] Samed Düzlü[2] **Rune Fiedler**[2] Marc Fischlin[2] Christian Janson[2]

[1]CISPA, Germany
cremers@cispa.de

[2]TU Darmstadt, Germany
samed@qpc.tu-darmstadt.de, {rune.fiedler, marc.fischlin, christian.janson}@cryptoplexity.de

3rd NIST PQC Standardization Conference

Are signatures exclusively owned by one public key?

- Possible unexpected behavior of unforgeable signature schemes:
  Given a signature, generate a new public key under which the signature verifies
- Property preventing this:
  Exclusive Ownership (S-DEO) [PS05, JCCS19, BCJZ21, CGN20]

Is a signature bound to a message?

- Possible unexpected behavior of unforgeable signature schemes:
  Compute a public key for which a signature can verify several messages
- Property preventing this:
  Message-Bound Signatures (MBS) [JCCS19, BCJZ21, CGN20]

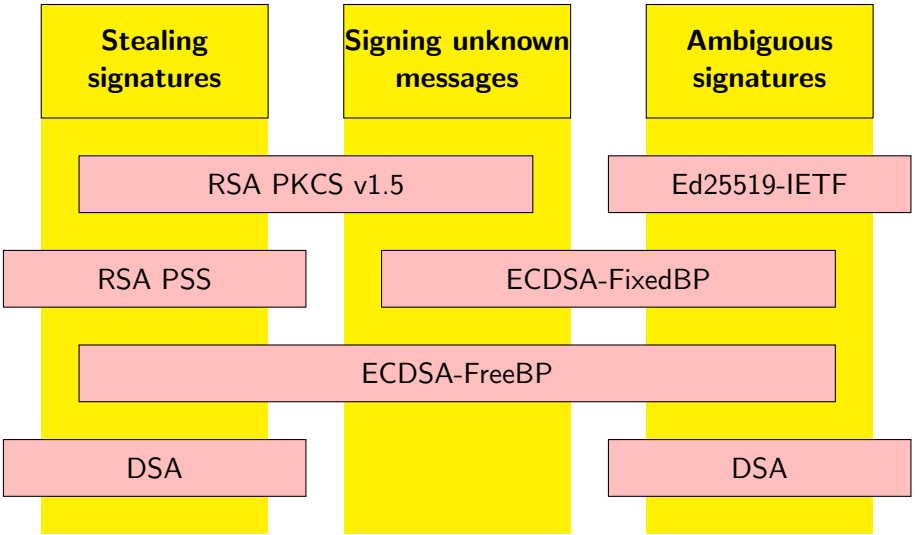
Can you sign an unknown message?

- Possible unexpected behavior of unforgeable signature schemes:
  Given a signature but not its message, produce a new public key under which the same message verifies
- Property preventing this:
  Non re-signability (NR) [JCCS19]

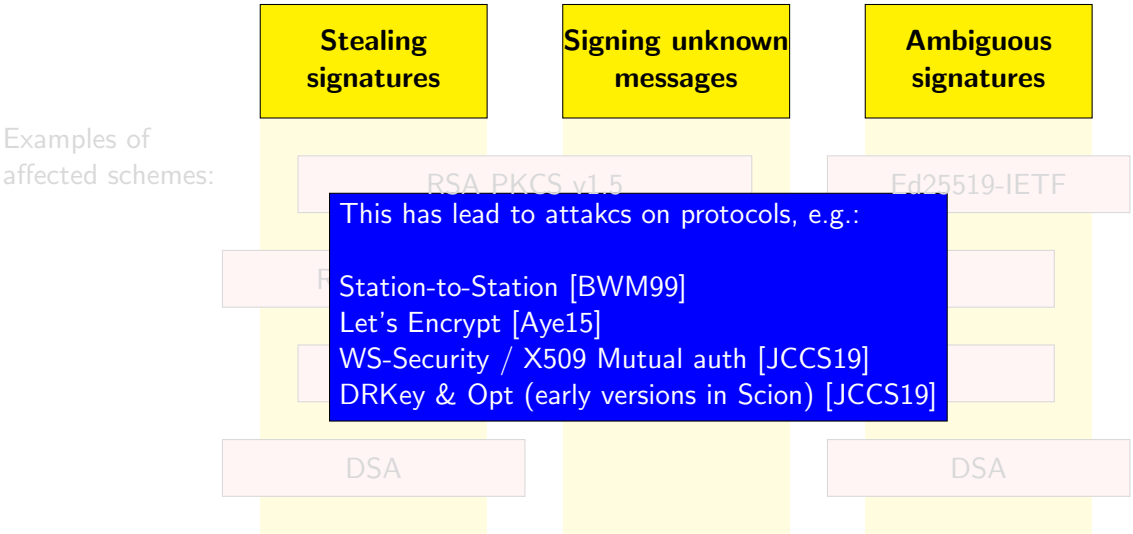
Unforgeability does not protect from malicious public keys

Unforgeability does not protect from malicious public keys

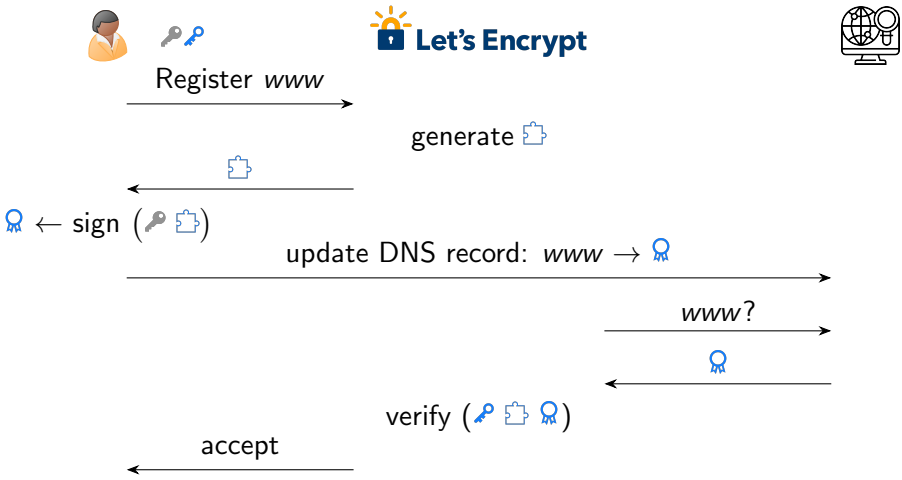
Examples of affected schemes:



Unforgeability does not protect from malicious public keys

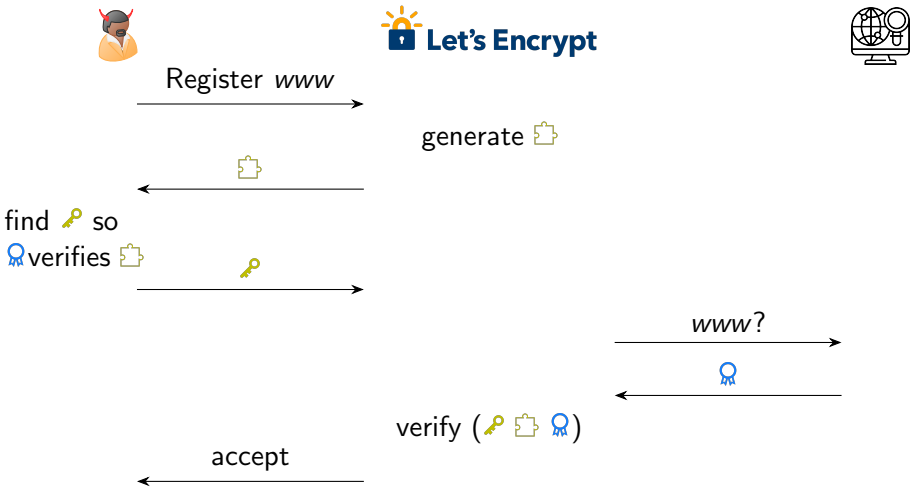


Attacking real-world protocols with malicious public keys I



Attack on Let's Encrypt ACME Draft 00 Protocol by [JCCS19]

Attacking real-world protocols with malicious public keys II



Beyond UnForgeability Features (BUFF)

- Schemes that offer S-CEO/S-DEO, MBS, and NR provide BUFF:

Beyond
Un**F**orgeability
Features

- and don't have the unexpected behaviors of stealing signatures, signing unknown messages, or ambiguous signatures.

Several NIST finalists lack BUFF

	Scheme	S-CEO & S-DEO	MBS	NR
main	CRYSTALS-Dilithium	✓	✓	✓
main	FALCON	✗	✓	✗
main	Rainbow Standard	✗	✓	✗
main	Rainbow CZ & Compr.	✦	✓	✗
alternate	GeMSS	✗	✗	✗
alternate	Picnic	✓	✓	✓
alternate	SPHINCS$^+$	✦	✓	✦

✓ property holds ✗ attack given ✦ inconclusive

Our generic BUFF transformation

- Schemes can be transformed to meet BUFF by adding scheme-specific checks
- or by applying our generic BUFF transformation:
  - Compute $\mathsf{H}(\mathsf{pk}, m)$. Sign digest. Prepend digest to the signature.

    $h \leftarrow \mathsf{H}(\mathsf{pk}, m)$
    $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, h)$
    return (h, σ)
  - Verification additionally checks the digest

    $h = \mathsf{H}(\mathsf{pk}, m)$
    $\wedge\ \mathsf{Vf}(\mathsf{pk}, h, \sigma)$

Our generic BUFF transformation achieves BUFF

$h \leftarrow \mathsf{H}(\mathsf{pk}, m)$
$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, h)$
return (h, σ)

$h = \mathsf{H}(\mathsf{pk}, m)$
$\wedge\ \mathsf{Vf}(\mathsf{pk}, h, \sigma)$

- Hashing in Sign binds to (pk, m)
- Checking digest in Vf prevents weak keys (where Vf always returns true)
- Formally, security reduces to security properties of H

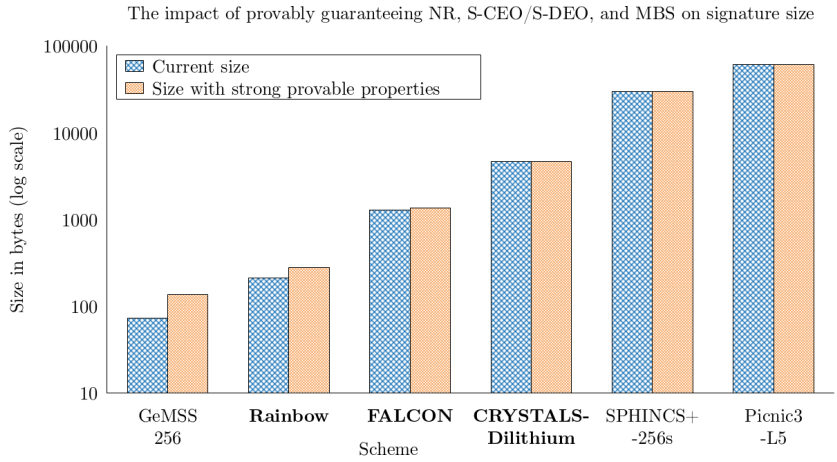
Our generic BUFF transformation is efficient

$h \leftarrow \mathsf{H}(\mathsf{pk}, m)$
$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, h)$
return (h, σ)

$h = \mathsf{H}(\mathsf{pk}, m)$
$\wedge\ \mathsf{Vf}(\mathsf{pk}, h, \sigma)$

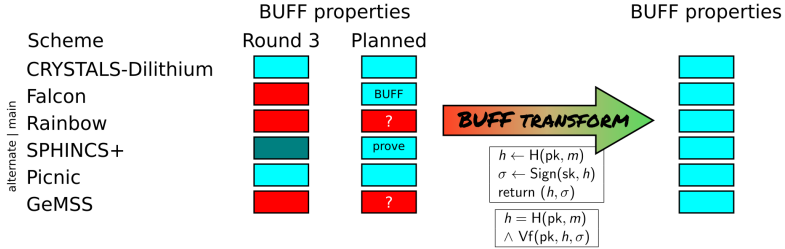
- One hash function evaluation in Sign and Vf each
- Signature size increases by the size of one hash digest

BUFF transformation keeps relative signature sizes



BUFF finalists now, prevent headaches later!

- Protect upcoming standard against maliciously generated public keys!
- Scheme-specific proofs or apply generic BUFF transformation
- Situation similar to length-extension resilience of SHA3
- NIST chooses Dilithium or FALCON (and SPHINCS+), all three will have BUFF!



Full paper: `https://eprint.iacr.org/2020/1525` (and IEEE S&P 2021)

rune.fiedler@cryptoplexity.de

References I

[Aye15] Andrew Ayer.
Duplicate Signature Key Selection Attack in Let's Encrypt, 2015.
https://www.agwa.name/blog/post/duplicate_signature_key_selection_attack_in_lets_encrypt.

[BCJZ21] Jacqueline Brendel, Cas Cremers, Dennis Jackson, and Mang Zhao.
The provable security of Ed25519: Theory and practice.
In *IEEE S&P (to appear)*, 2021.
https://eprint.iacr.org/2020/823.

[BWM99] Simon Blake-Wilson and Alfred Menezes.
Unknown key-share attacks on the station-to-station (STS) protocol.
In *PKC*, 1999.
https://link.springer.com/chapter/10.1007/978-3-642-19074-2_18.

[CGN20] Konstantinos Chalkias, François Garillot, and Valeria Nikolaenko.
Taming the Many EdDSAs.
In *SSR*, 2020.
https://eprint.iacr.org/2020/1244.

References II

[JCCS19] Dennis Jackson, Cas Cremers, Katriel Cohn-Gordon, and Ralf Sasse.
Seems legit: Automated analysis of subtle attacks on protocols that use signatures.
In *ACM CCS*, 2019.
https://eprint.iacr.org/2019/779.

[PS05] Thomas Pornin and Julien P. Stern.
Digital signatures do not guarantee exclusive ownership.
In *ACNS*, 2005.
https://link.springer.com/chapter/10.1007/11496137_10.

Picture references

- secret key icon by Yannick Lung
- public key icon by Yannick Lung
- "Let's Encrypt Wide" by Let's Encrypt is licensed under CC BY-NC 4.0
- DNS icon made by Eucalyp from Flaticon
- puzzle icon by Becris.
- Signature icon by PINPOINT.WORLD is licensed under CC BY 3.0
- Message icon by Yannick Lung

Fiat-Shamir Transform implements BUFF transform

Sign(sk, m) :

$c \leftarrow \mathsf{H}(\mathsf{pk}, m, a)$

$\dots$

$\sigma \leftarrow (c, \dots)$

- Dilithium (LWE, SIS)
- Picnic (ZK, MPC-in-the-head)

- Fiat-Shamir transform implements BUFF transform
- Dilithium and Picnic provide BUFF

FALCON

$\mathsf{Vf}_{FALCON}(\mathsf{pk} = h, m, \sigma = (r, s_2))$

$c \leftarrow \mathsf{H}(r, m)$

$s_1 \leftarrow c - s_2 h$

$d \leftarrow \|(s_1, s_2)\|^2 \leq \lfloor \beta^2 \rfloor$

$\downarrow$

d

- leverage (non-)invertibility of s_2 to break exclusive ownership and NR

Rainbow (& GeMSS)

$\mathsf{Vf}_{Rainbow}(\mathsf{pk} = \mathcal{P}, m, \sigma = (\mathsf{z}, r))$

$h \leftarrow \mathsf{H}(\mathsf{H}(m), r)$

$d \leftarrow \mathcal{P}(\mathsf{z}) = h$

$\downarrow$

d

- Construct tailored public map $\mathcal{P}$ for fixed digest to break exclusive ownership
- Re-sign digest under own key to break NR
- GeMSS has additional input to $\mathcal{P}$ that allows wiggle room to break MBS

SPHINCS+

$\mathsf{Vf}_{SPHINCS+}(\mathsf{pk} = (seed, root), m, \sigma = (r, \sigma_{HT}))$

$digest \leftarrow \mathsf{H}_{msg}(r, \mathsf{pk}.seed, \mathsf{pk}.root, m)$

$root' \leftarrow$ hash to the root$(digest, \mathsf{pk}.seed, \sigma_{HT})$

$d \leftarrow root' = \mathsf{pk}.root$

$\downarrow$

d

- Breaking exclusive ownership requires finding $\mathsf{pk}'.root = \mathsf{H}(...\mathsf{H}_{msg}(\mathsf{pk}'.root, ...))$
- If signature leaks *digest*, NR reduces to ΦNM of H_{msg}, otherwise attacker can only guess

Comparison of transformations

Transform.	Signature	S-CEO	S-DEO	M-S-UEO	MBS	NR
[PS05]-1	$\mathsf{Sig}(\mathsf{sk}, m)$, $\mathsf{H}(m)$	✗	✓	✗	✓	✗
[PS05]-2	$\mathsf{Sig}(\mathsf{sk}, m)$, $\mathsf{H}(\mathsf{pk})$	✓	✓	✓	✗	✗
[PS05]-3	$\mathsf{Sig}(\mathsf{sk}, \mathsf{H}(m, \mathsf{pk}))$	✗ (✓)	✗ (✓)	✗	✗	✗
BUFF	$\mathsf{Sig}(\mathsf{sk}, \mathsf{H}(m, \mathsf{pk}))$, $\mathsf{H}(m, \mathsf{pk})$	✓	✓	✓	✓	✓

✓ provides property ✗ vulnerable (✓) provides property if no weak keys