Block Cipher Modes of Operation: Provable Security Using Automated Reasoning

Nicky Mouha

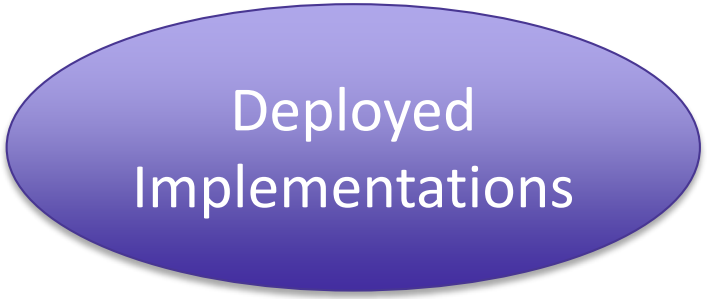
GAPS 2025

September 5, 2025

Applications of Automated Reasoning

Is the math / proof correct?

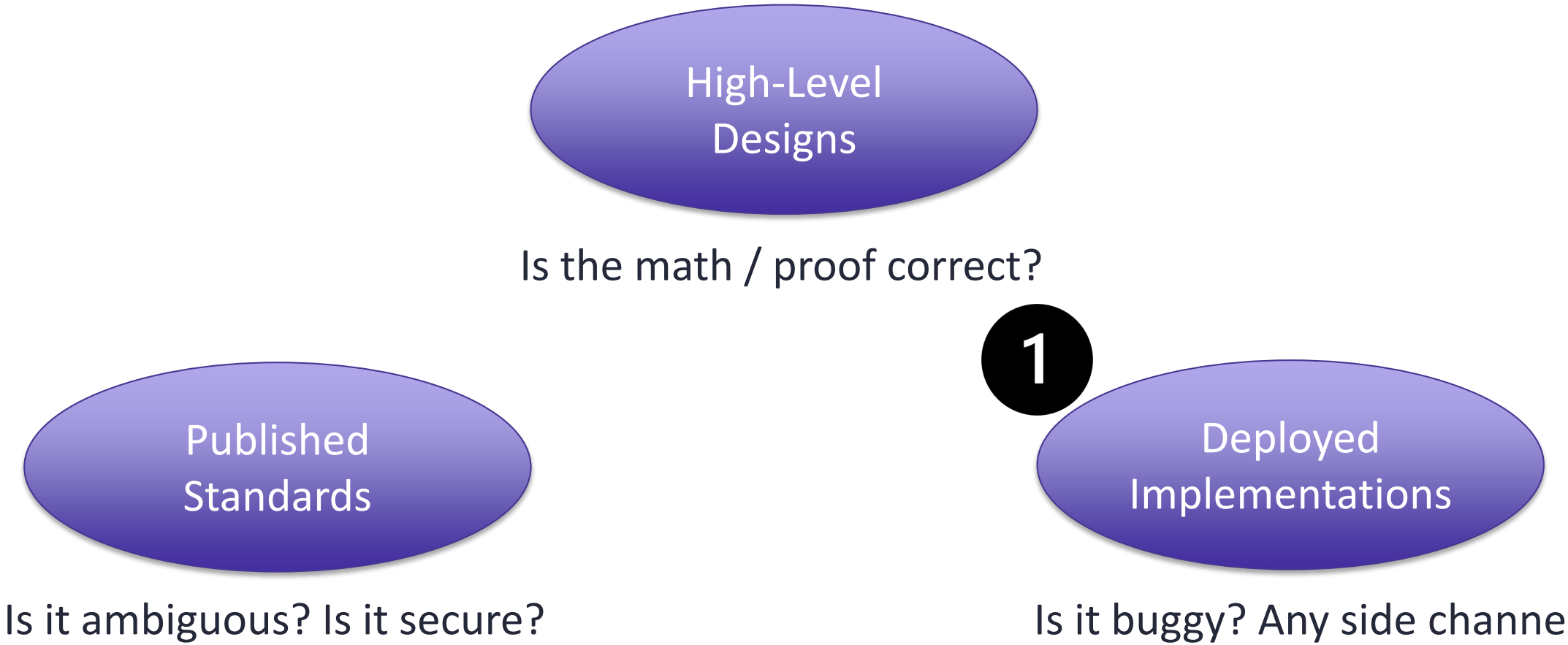
Is it ambiguous? Is it secure?



Is it buggy? Any side channels?

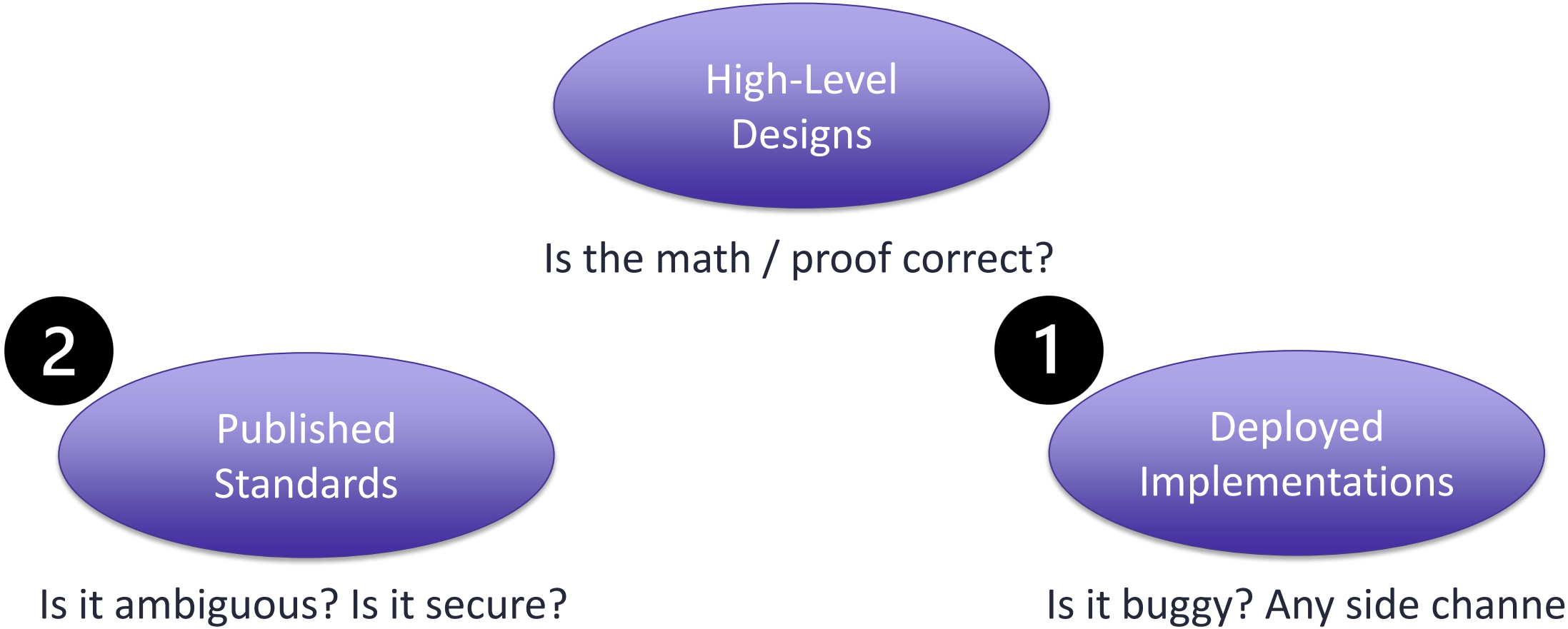
Slide: Invited Talk by Karthik Bhargavan (CRYPTO 2024)

Applications of Automated Reasoning



Slide: Invited Talk by Karthik Bhargavan (CRYPTO 2024)

Applications of Automated Reasoning

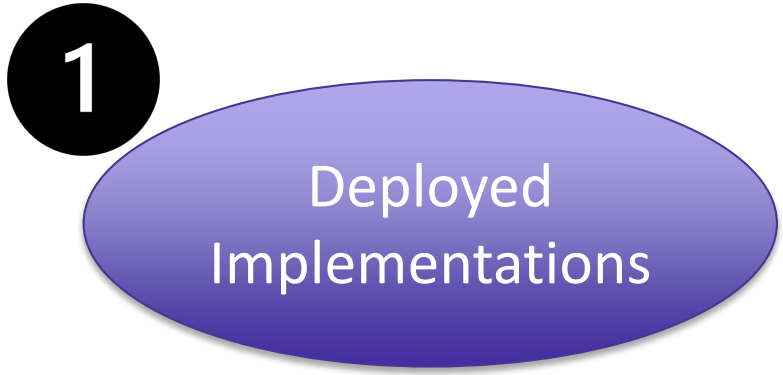


Slide: Invited Talk by Karthik Bhargavan (CRYPTO 2024)

Applications of Automated Reasoning

Is the math / proof correct?

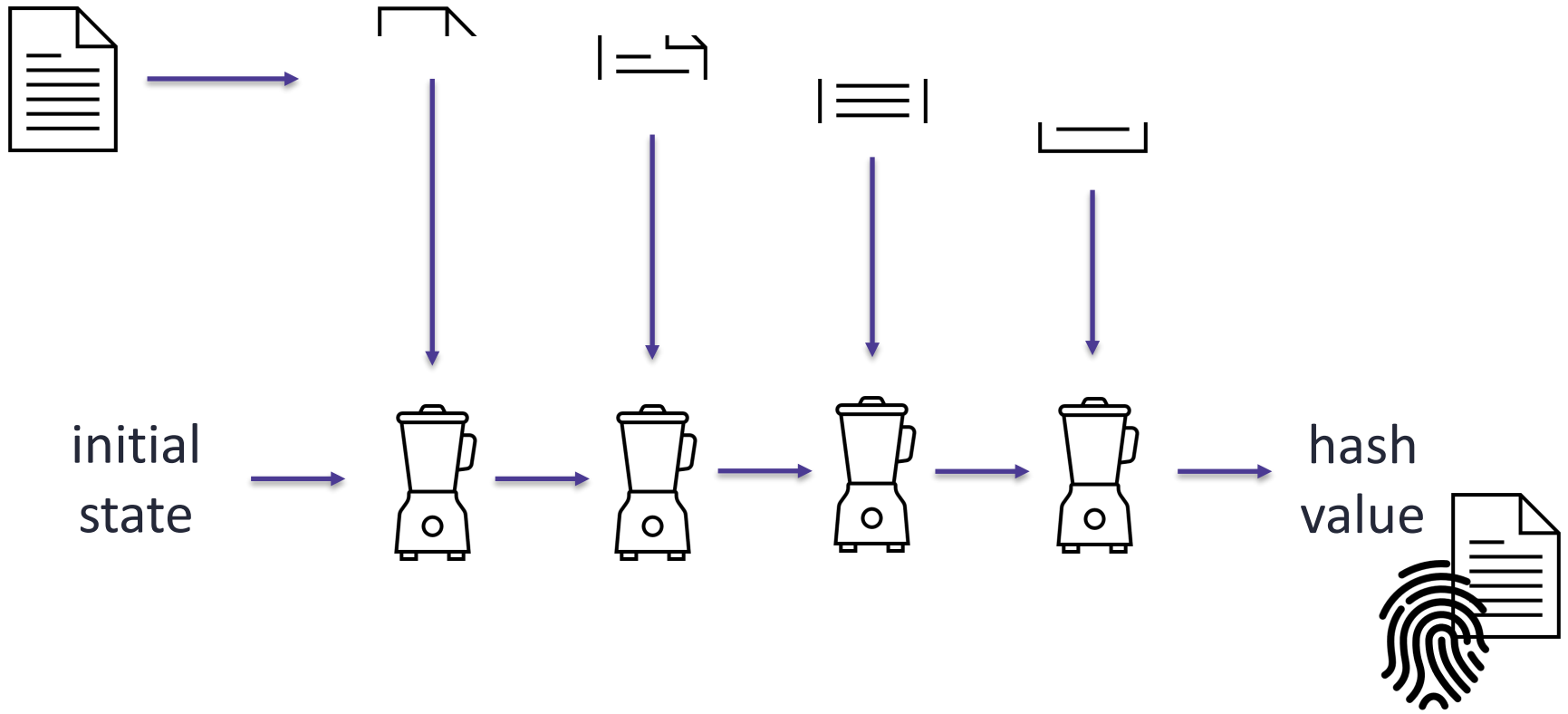
Is it ambiguous? Is it secure?



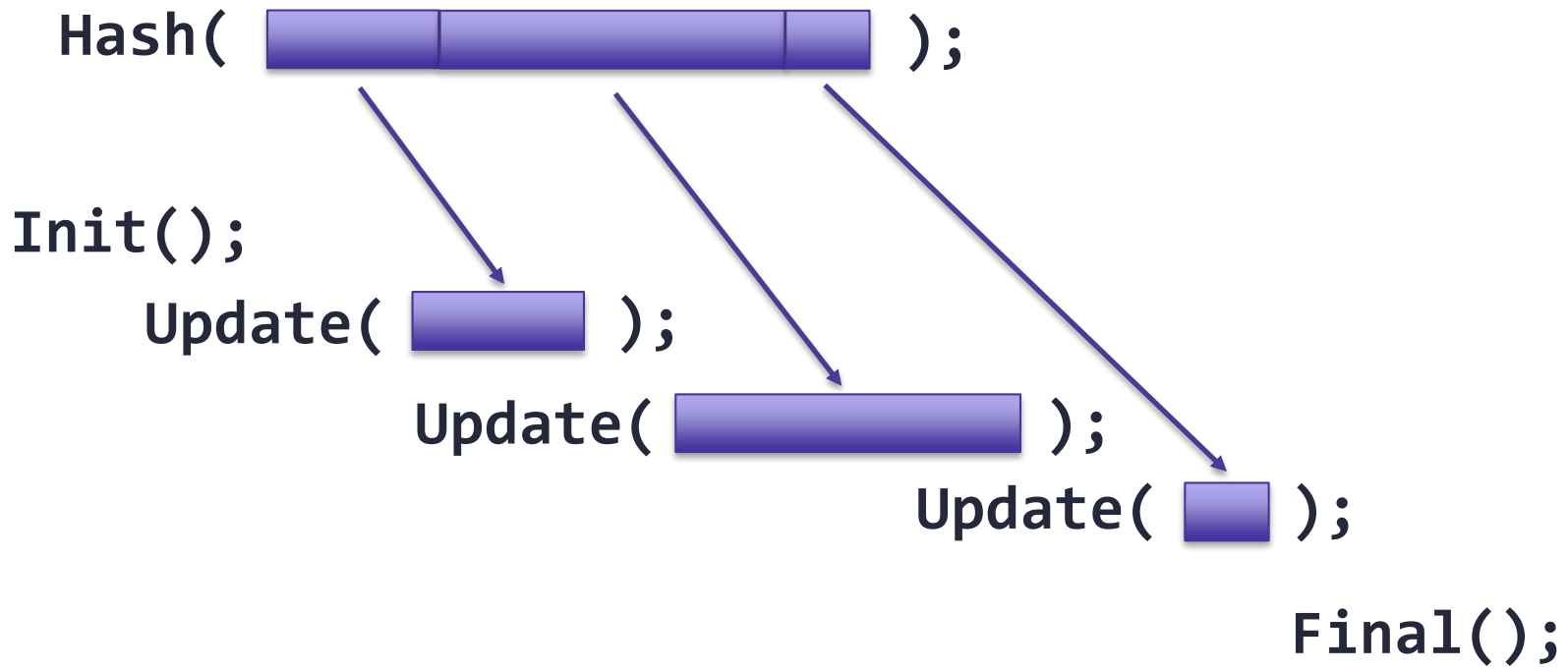
Is it buggy? Any side channels?

Slide: Invited Talk by Karthik Bhargavan (CRYPTO 2024)

Iterated Hash Functions



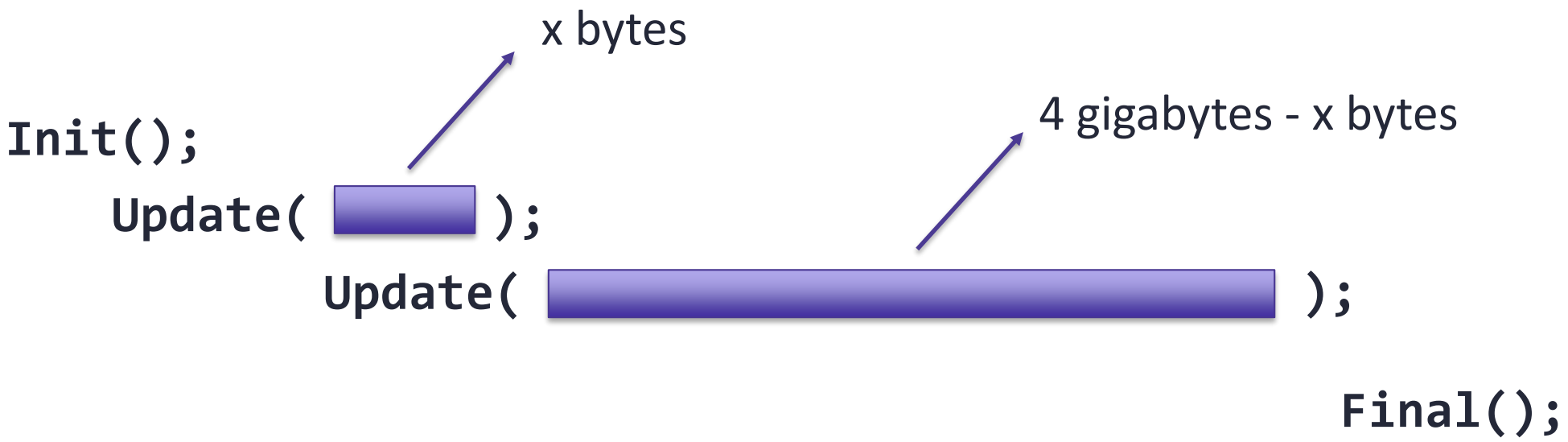
Two Common Hash Function Interfaces



- Q: Where/when are they used?

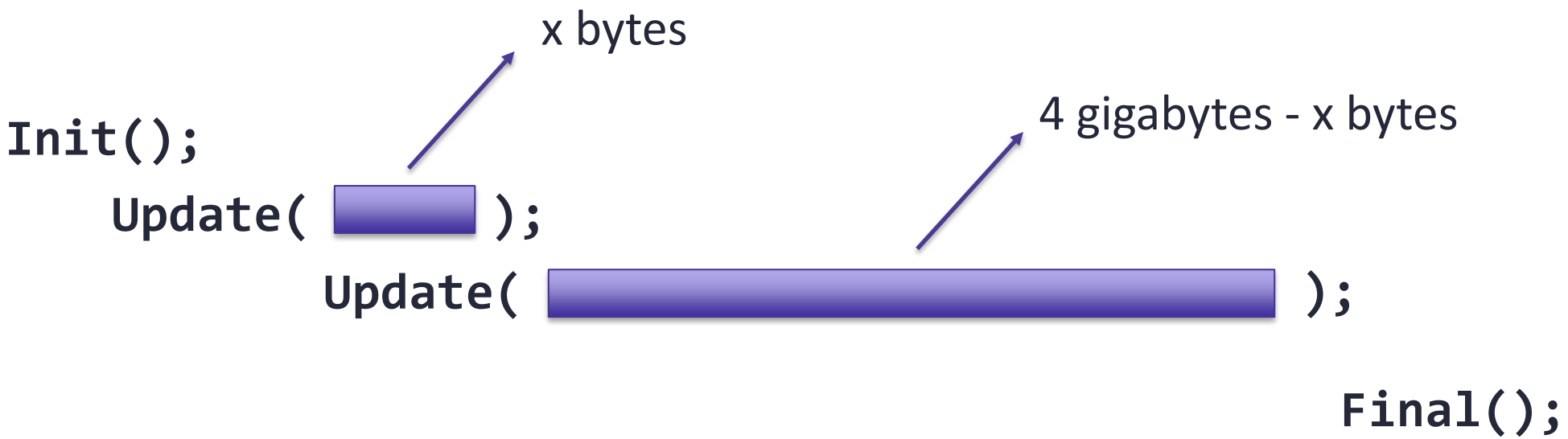
SHA-3 Bug (CT-RSA 2023)

- Appeared in 2011 (final-round Keccak submission)
- CVE-2022-37454, NVD: 9.8 CRITICAL



SHA-3 Bug (CT-RSA 2023)

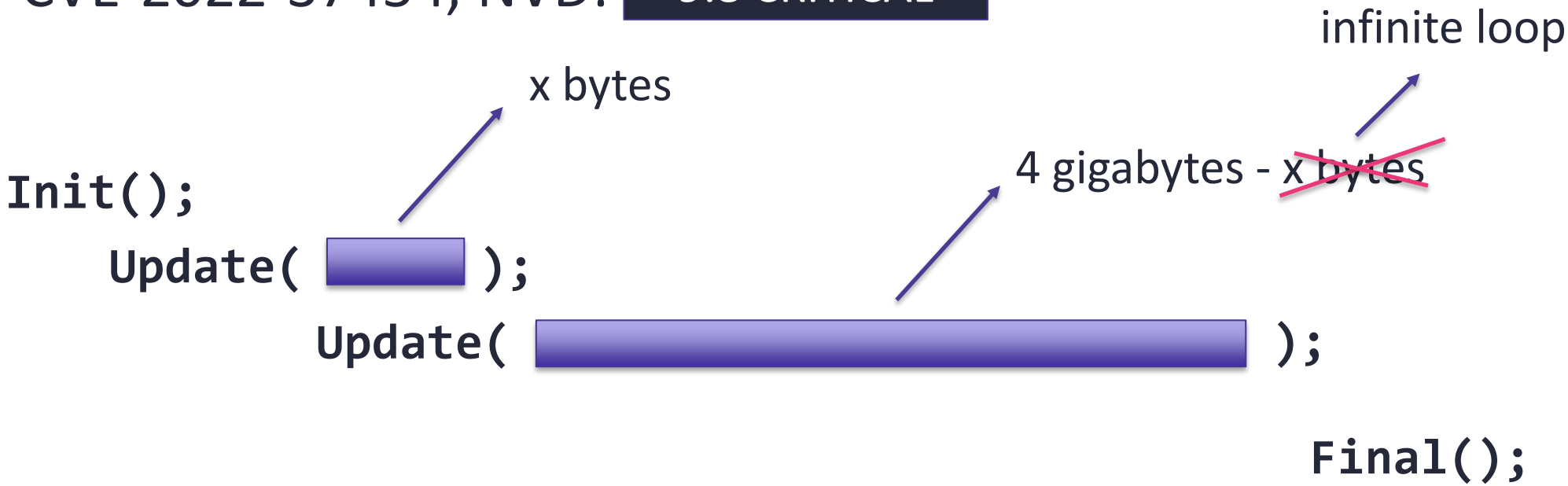
- Appeared in 2011 (final-round Keccak submission)
- CVE-2022-37454, NVD: 9.8 CRITICAL



buffer overflow!
(arbitrary code execution)

SHA-3 Bug (CT-RSA 2023)

- Appeared in 2011 (final-round Keccak submission)
- CVE-2022-37454, NVD: 9.8 CRITICAL



buffer overflow!
(arbitrary code execution)

KeccakSponge.inc

```
partialBlock = (unsigned int) (dataByteLen - i);
if (partialBlock + instance->byteIOIndex > rateInBytes) {
    partialBlock = rateInBytes - instance->byteIOIndex;
}
```

KeccakSponge.inc

```
partialBlock = (unsigned int) (dataByteLen - i);
if (partialBlock > rateInBytes - instance->byteIOIndex) {
    partialBlock = rateInBytes - instance->byteIOIndex;
}
```

KeccakSponge.inc

```
partialBlock = (unsigned int) (dataByteLen - i);
if (dataByteLen - i > rateInBytes - instance->byteIOIndex) {
    partialBlock = rateInBytes - instance->byteIOIndex;
}
```

KeccakSponge.inc

```
if (dataByteLen - i > rateInBytes - instance->byteIOIndex) {
    partialBlock = rateInBytes - instance->byteIOIndex;
} else {
    partialBlock = (unsigned int) (dataByteLen - i);
}
```

CodeQL

"Add query for CVE-2022-37454" (https://github.com/github/codeql/pull/12036)

```
todo = digest_len;
if (done + todo > out_len) {
  todo = out_len - done;
}
OPENSSL_memcpy(out_key + done, previous, todo);
done += todo;
```

CodeQL

"Add query for CVE-2022-37454" (https://github.com/github/codeql/pull/12036)

```
todo = digest_len;
if (todo > out_len - done) {
  todo = out_len - done;
}
OPENSSL_memcpy(out_key + done, previous, todo);
done += todo;
```

Unclear/Ambiguous Specifications

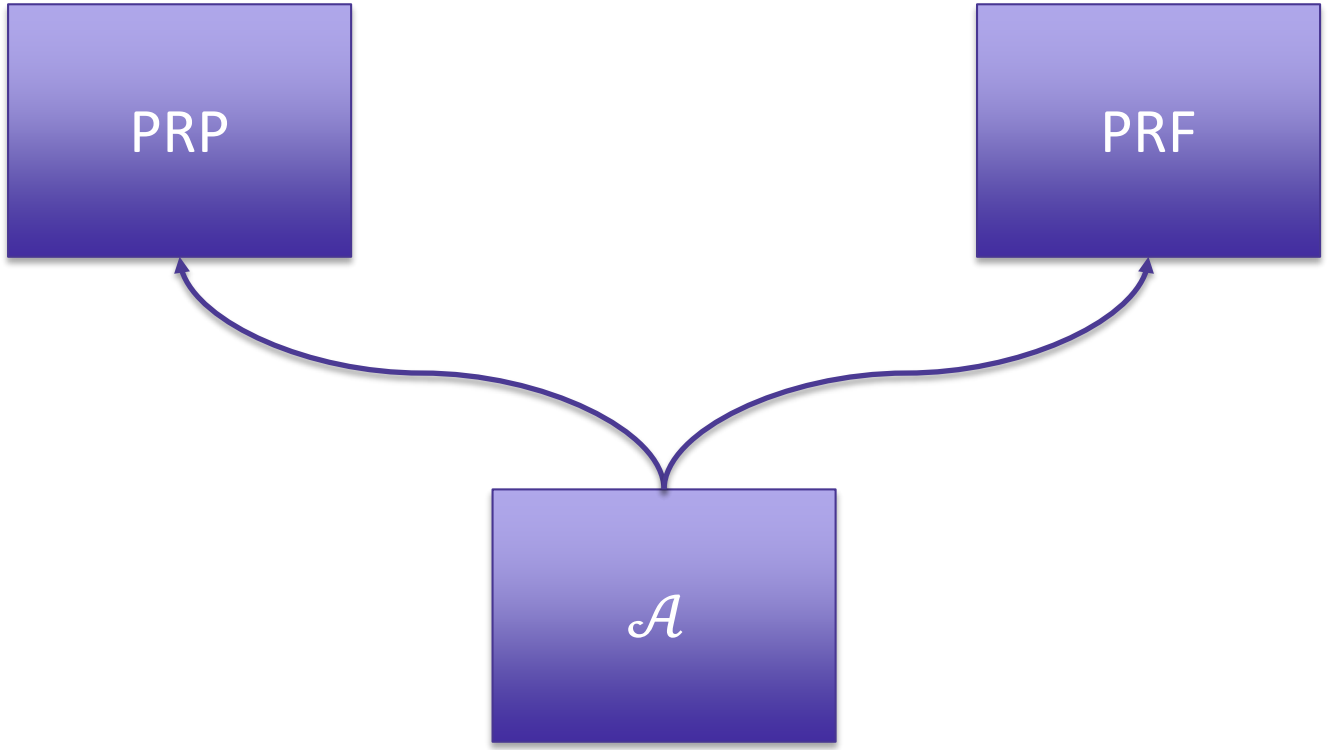
- OpenSSL HMAC API (Benmocha, et al., SAC 2021)

Unclear/Ambiguous Specifications

- OpenSSL HMAC API (Benmocha, et al., SAC 2021)

- OpenSSL zeroization vulnerability (Olmos et al., CHES 2024)
  - Not vulnerability according to maintainers, OpenSSL doesn't claim security in this model

PRP-PRF Switching Lemma



$$|\Pr[\mathcal{A}^{\mathrm{PRP}} \to 1] - \Pr[\mathcal{A}^{\mathrm{PRF}} \to 1]| \leq \binom{q}{2}\frac{1}{2^n} = \frac{q(q-1)}{2^{n+1}}$$

Incorrect Proof and Fix

- Error in PRP-PRF Switching Lemma Proof
  - "Code-Based Game-Playing Proofs and the Security of Triple Encryption" (Bellare and Rogaway, ePrint 2004/331)

- How to fix?
  - Game-playing proofs
  - Patarin's H-Coefficient Technique ⟵ this talk

Patarin's H-Coefficient Technique

- Transcript τ summarizes interaction with oracles
  - Probability distribution of τ in the real (resp. ideal) world: X (resp. Y)
  - Transcript τ is attainable in real world: $\Pr[X = \tau] > 0$

- Set of attainable transcripts: $\mathcal{T}_{\text{good}} \cup \mathcal{T}_{\text{bad}}$

- Let ε be such that for all $\tau \in \mathcal{T}_{\text{good}}$: $\frac{\Pr[X=\tau]}{\Pr[Y=\tau]} \geq 1 - \varepsilon$
  - Then $\text{Adv}(\mathcal{A}) \leq \varepsilon + \Pr(Y \in \mathcal{T}_{\text{bad}})$

Chaskey: Security Proof

Click for demo

$$(N - D_1)! \cdot (N - D_2)! \cdot (N - D_3)! \cdot (N - T)!$$
$$\leq$$
$$(N - D_1 - D_2 - D_3 - T)! \cdot (N!)^3$$

https://eprint.iacr.org/2014/386.pdf (page 11)

Chaskey: Security Proof

Click for demo

easy

$$(N - D_1)! \cdot (N - D_2)! \cdot (N - D_3)! \cdot (N - T)!$$
$$\leq$$
$$(N - D_1 - D_2 - D_3 - T)! \cdot (N!)^3$$

https://eprint.iacr.org/2014/386.pdf (page 11)

Chaskey: Security Proof

Click for demo

???

$$(N - D_1)! \cdot (N - D_2)! \cdot (N - D_3)! \cdot (N - T)!$$
$$\leq$$
$$(N - D_1 - D_2 - D_3 - T)! \cdot (N!)^3$$

https://eprint.iacr.org/2014/386.pdf (page 11)

Chaskey: Falling Factorial

Click for demo

$$(N - D_1)! \cdot (N - D_2)! \cdot (N - D_3)! \cdot (N - T)!$$
$$\leq$$
$$(N - D_1 - D_2 - D_3 - T)! \cdot (N!)^3$$

Let $(x)_n = \underbrace{x(x-1)(x-2)\cdots(x-n+1)}_{n \text{ factors}}$

$$(N - T)_{D_1 + D_2 + D_3}$$
$$\leq$$
$$(N)_{D_1} \cdot (N)_{D_2} \cdot (N)_{D_3}$$

Chaskey: Induction?

Click for demo

Click for demo

Proof by induction on $N - T$ requires:

$$(N + 1 - D_1) \cdot (N + 1 - D_2) \cdot (N + 1 - D_3) \cdot (N - T + 1)$$
$$\leq$$
$$(N - T + 1 - D_1 - D_2 - D_3) \cdot (N + 1)^3$$

Chaskey: Induction?

Click for demo

Click for demo

Proof by induction on $N - T$ requires:

$$(N + 1 - D_1) \cdot (N + 1 - D_2) \cdot (N + 1 - D_3) \cdot (N - T + 1)$$

$$\geq$$

$$(N - T + 1 - D_1 - D_2 - D_3) \cdot (N + 1)^3$$

But: inequality holds in other direction!

Chaskey: Correct Proof

Click for demo

Theorem descFactorial_mul_descFactorial: $(n)_m = (n)_k \cdot (n-k)_{m-k}$

$$
\begin{aligned}
&(N-T)_{D_1+D_2+D_3} \\
&= (N-T)_{D_1} \cdot (N-T-D_1)_{D_2+D_3} \\
&= (N-T)_{D_1} \cdot (N-T-D_1)_{D_2} \cdot (N-T-D_1-D_2)_{D_3}
\end{aligned}
$$

Theorem descFactorial_le: $k \leq m \rightarrow (k)_n \leq (m)_n$

$$
\begin{aligned}
&(N-T)_{D_1} \cdot (N-T-D_1)_{D_2} \cdot (N-T-D_1-D_2)_{D_3} \\
&\leq \\
&(N)_{D_1} \cdot (N)_{D_2} \cdot (N)_{D_3}
\end{aligned}
$$

Lots of Math Theorems Needed!

- Formalizing 100 Theorems: https://cs.ru.nl/~freek/100/
- Proofs for 99 Theorems
- Theorem 10 (Fermat's Last Theorem): in progress

Conclusion

- Is the implementation buggy?
- Is the specification clear?
- Is the proof correct?

Conclusion

- Is the implementation buggy?
- Is the specification clear?
- Is the proof correct?

Questions?