CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

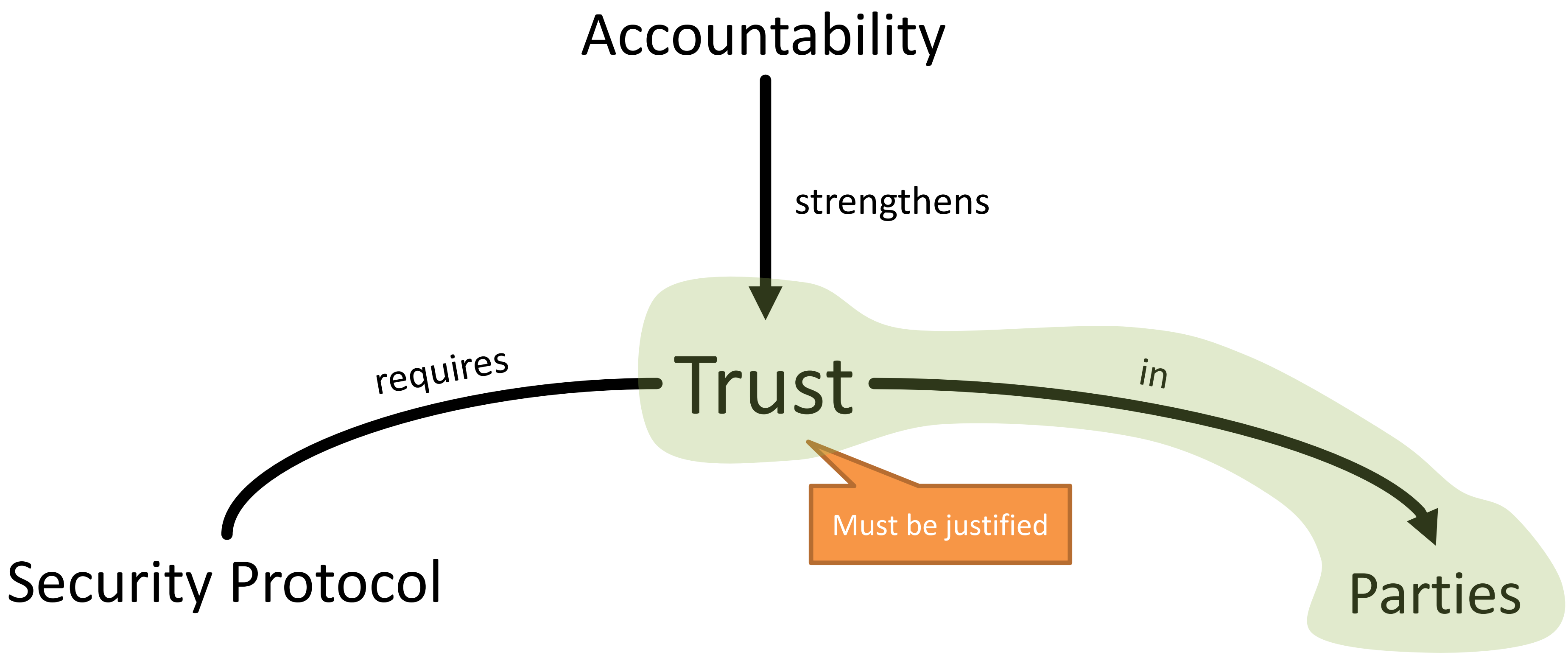
Verifying Accountability for Unbounded Sets of Participants

34th IEEE Computer Security Foundations Symposium

Kevin Morio, Robert Künnemann

CISPA Helmholtz Center for Information Security
kevin.morio@cispa.de, robert.kuennemann@cispa.saarland

Trust



Accountability
strengthens
requires
Trust
in
Security Protocol
Must be justified
Parties

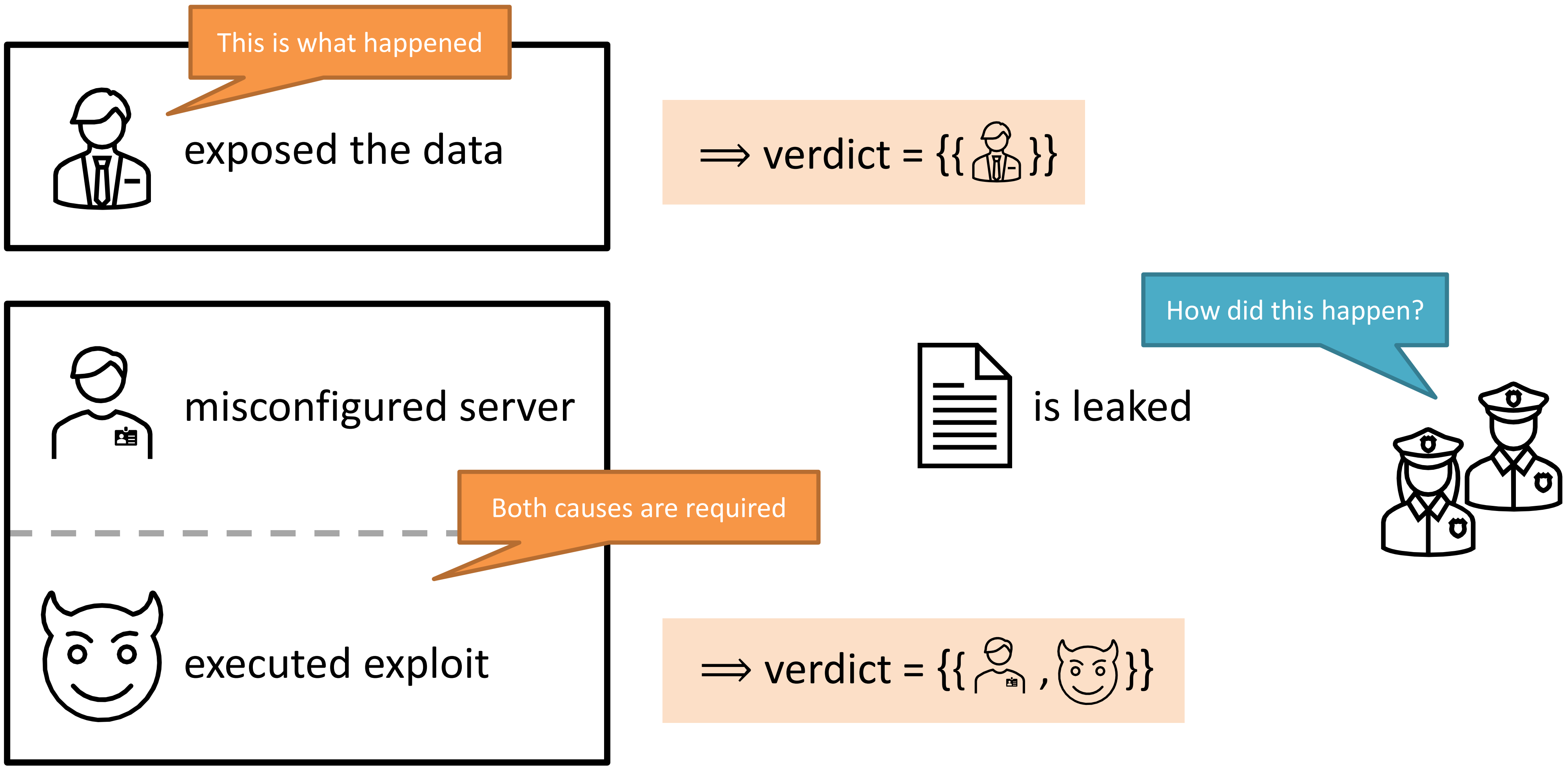
Accountability by Causation

- Accountability notion of Künneman et. al. (2019)
- Based on sufficient causation

- **Accountability for φ**
  - Meta property of a protocol
  - Allows identifying all parties causing a violation of **φ**

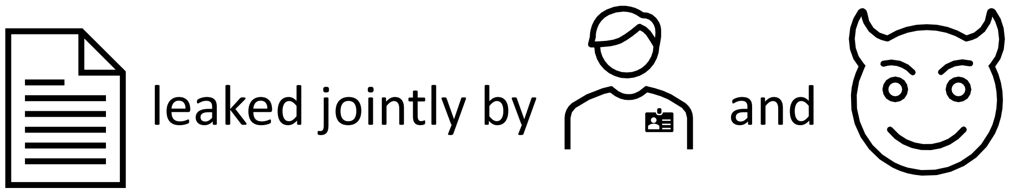
When is a party's behavior the cause of a violation?

Accountability by Causation



Counterfactual Relation

Actual Situation



Counterfactual Situation

Need relation between actual and counterfactual world!

Automated Verification of Accountability

- How can accountability be verified?

$\Longrightarrow$ $\mathrm{verdict}(t)$ **provides accountability for φ**

First Translation (CSF 2019)

- Case distinction on different verdicts

$$verdict(t) := \begin{cases} V_1 & \text{if } \omega_1(t) \\ \vdots & \\ V_n & \text{if } \omega_n(t) \end{cases}$$

- Cases are **exhaustive** and **exclusive**
- **Problems**

  1. Finite number of verdicts
  2. Verdicts must be stated explicitly

  $\Longrightarrow$ Only a bounded number of parties are supported

New Approach

- **Case tests:** Trace properties with free variables

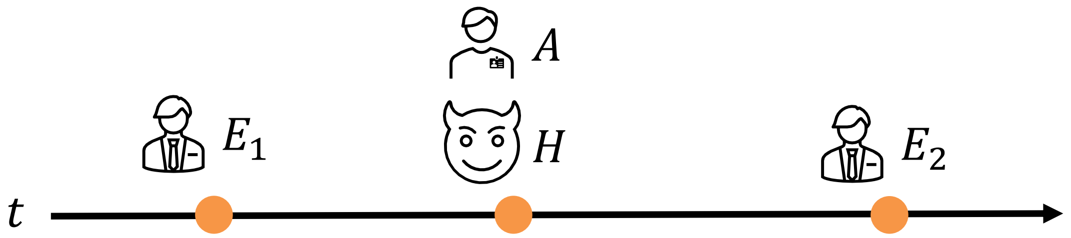
$$\tau_1 := \exists data, i.\ \mathsf{LeakEmployee}(e, data)@i$$
$$\tau_2 := \exists data, i.\ \mathsf{LeakAdminHacker}(a, h, data)@i$$

Free variables

- **Verdict function:** Union over instantiated case tests

$$verdict(t) := \bigcup_{\tau \in \mathsf{tests}} \{ fv(\tau)\rho \mid \exists \rho.\ t \vDash \tau\rho \}$$

- Example



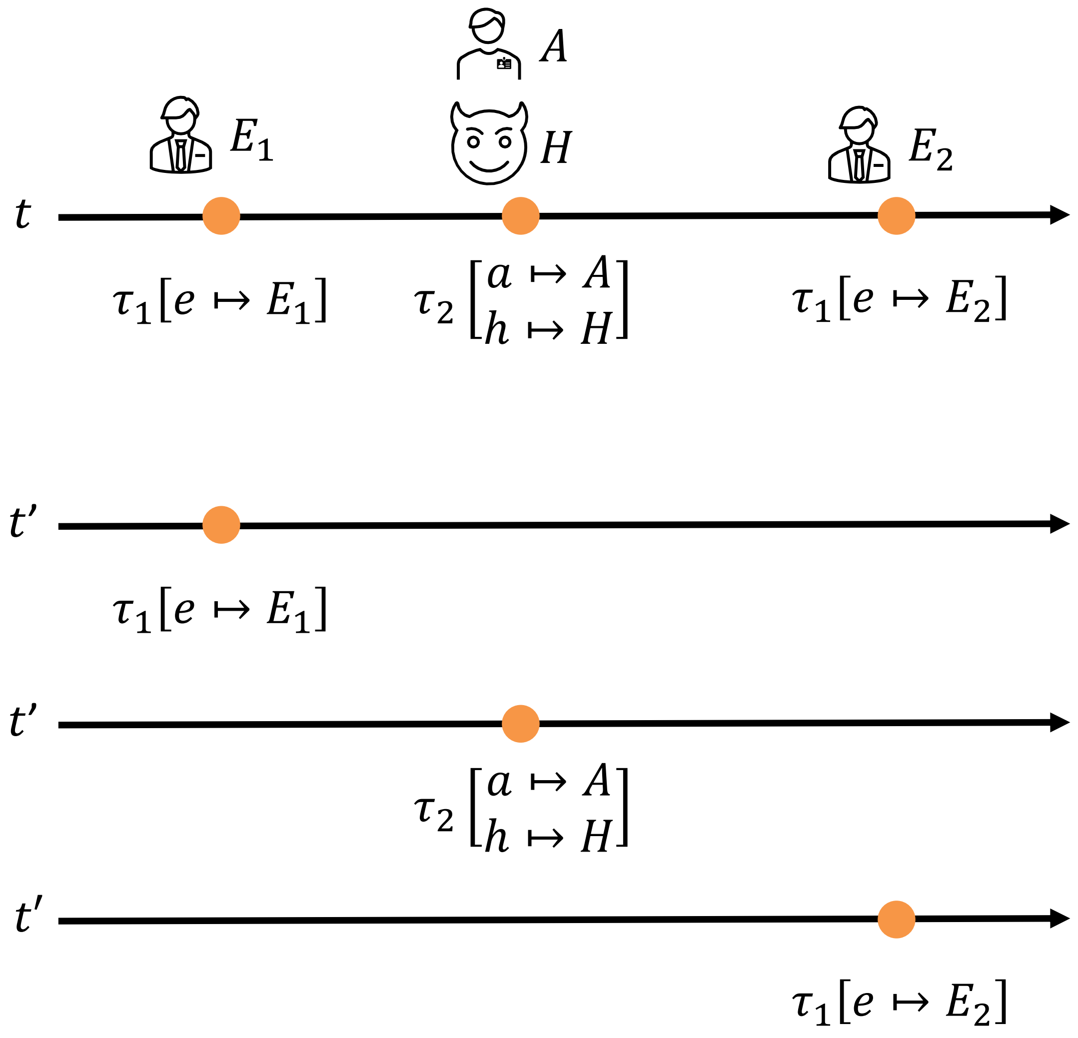
$\tau_1[e \mapsto E_1]$ $\quad$ $\tau_2\begin{bmatrix} a \mapsto A \\ h \mapsto H \end{bmatrix}$ $\quad$ $\tau_1[e \mapsto E_2]$

$$verdict(t) = \{ \{E_1\}, \{A, H\}, \{E_2\} \}$$

Challenge: Counterfactual Relation

Actual Trace

Counterfactual Traces



Implementation

- SAPiC / MSRs
- Case tests
- Accountability lemmas

↓ translate

- MSRs
- Standard lemmas

↓ Tamarin

- Verification
- Counterexample
- Timeout

```
test evidence:
    "Ex #i. Blame(m)@i"

lemma missing:
    evidence accounts for
    "All sid s ms #i. Send(<sid, s>, ms)@i
         ==> Ex m #j. Post(<sid, m>, '0', ms)@j"
```

```
lemma missing_evidence_suff: ...
/* ... */
lemma missing_evidence_single: ...
```

```
missing_evidence_suff: verified (16 steps)
```

```
missing_verif_empty: falsified - found trace (16 steps)
```

Case Studies

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

- 8 case studies (4 from prior work, 4 new)
- Prior work

	Our proposal			[21]		
WhoDunit (fixed)	✓	7	52 s	✓ (r_c)	8	24 s
				✓ (r_w)	7	11 s
Certificate Transparency (extended)	✓	27	17 s	✓	31	21 s
OCSP Stapling (trusted resp.)	✓	7	1 s	✓	7	515 s
OCSP Stapling (untrusted resp.)	✗	7	1 s	✗	7	75 s

- New case studies

Our proposal	1 role			2 roles			3 roles			4 roles			5 roles		
Basic DMN (duplicate ciphertexts)		—			—		✓	13	26 s		—			—	
DMN + message tracing (first)	✓	7	8 s	✓	7	124 s	✓	7	1373 s	✓	7	14 178 s	✓	7	134 160 s
DMN + message tracing (all)	✓	7	6 s	✗	7	12 s	✗	7	22 s	✗	7	100 s	✗	7	355 s
MixVote (unbounded)	✓	14	6 s		—			—			—			—	
[21]	1 party			2 parties			3 parties			4 parties			5 parties		
DMN + message tracing (first)	✓	7	7 s	✓	17	133 s	✓	46	2146 s	✓	149	23 827 s	—*	544	—
DMN + message tracing (all)	✓	7	4 s	✗	17	23 s	✗	46	115 s	✗	149	548 s	✗	544	2922 s
MixVote (unbounded)**	✓	14	5 s	✓	34	58 s	✓	92	2721 s	—*	298	—	—*	1112	—

* No verification results due to memory exhaustion. ** Each party acts in the same role, that of the server.

Conclusion

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

- Automated verification of accountability supporting an unbounded number of participants
  - Necessary for analyzing real-world protocols
- Case tests as the key concept
  - Flexible definition of verdict functions
  - Improved readability

- Implemented in Tamarin (github.com/kevinmorio/tamarin-prover)

- Up to 5x faster than the previous approach
- Less effort to formulate accountability lemmas