Secure Logging Schemes and Certificate Transparency

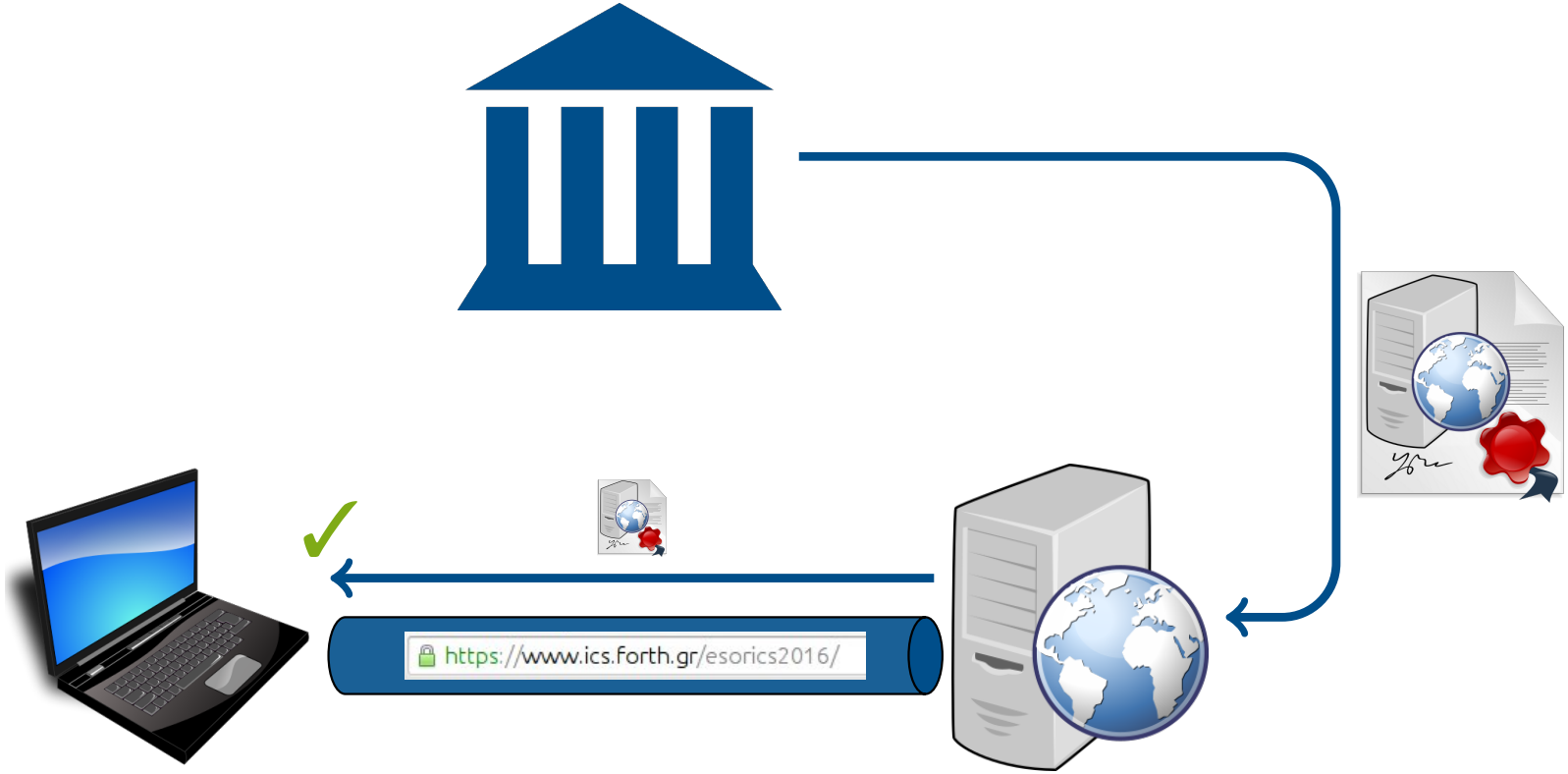
Felix Günther

Technische Universität Darmstadt, Germany

joint work with Benjamin Dowling, Udyani Herath, and Douglas Stebila

Secure Communication Requires Trust

TECHNISCHE UNIVERSITÄT DARMSTADT

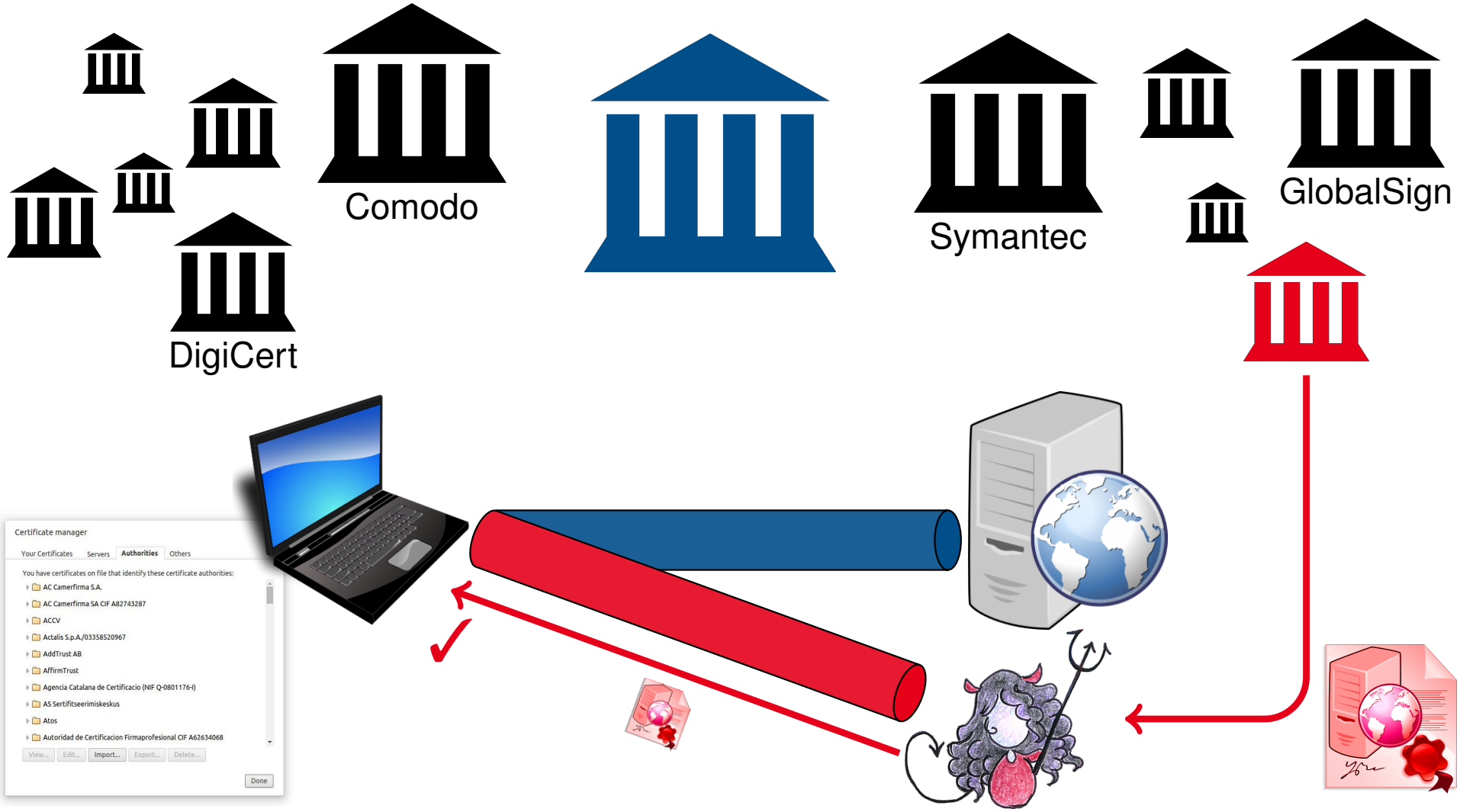


confidentiality **authenticity**

Many Certificate Authorities

. . . but single point of failure

TECHNISCHE UNIVERSITÄT DARMSTADT



A Severe Real-World Threat

TECHNISCHE UNIVERSITÄT DARMSTADT

DigiNotar, 2011

- intruder issued valid certificate for `google.com` + subdomains, Facebook, ...
- potentially active for several weeks before detection

Comodo Group, 2011

- nine fraudulent certificates
- for domains by Google, Yahoo!, Skype, and others

TURKTRUST, 2013

- mistakenly issued two intermediate CA certificates (instead of regular)
- issue remained undetected for over two years

...

Mitigation Approaches

TECHNISCHE UNIVERSITÄT DARMSTADT

Key Pinning

DANE

Certificate Transparency

...

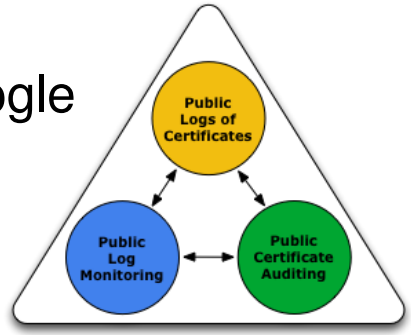
Revocation

OCSP

Certificate Transparency

TECHNISCHE UNIVERSITÄT DARMSTADT

- experimental IETF standard (RFC 6962) proposed by Google



- public logging of certificates
- open auditing and monitoring system

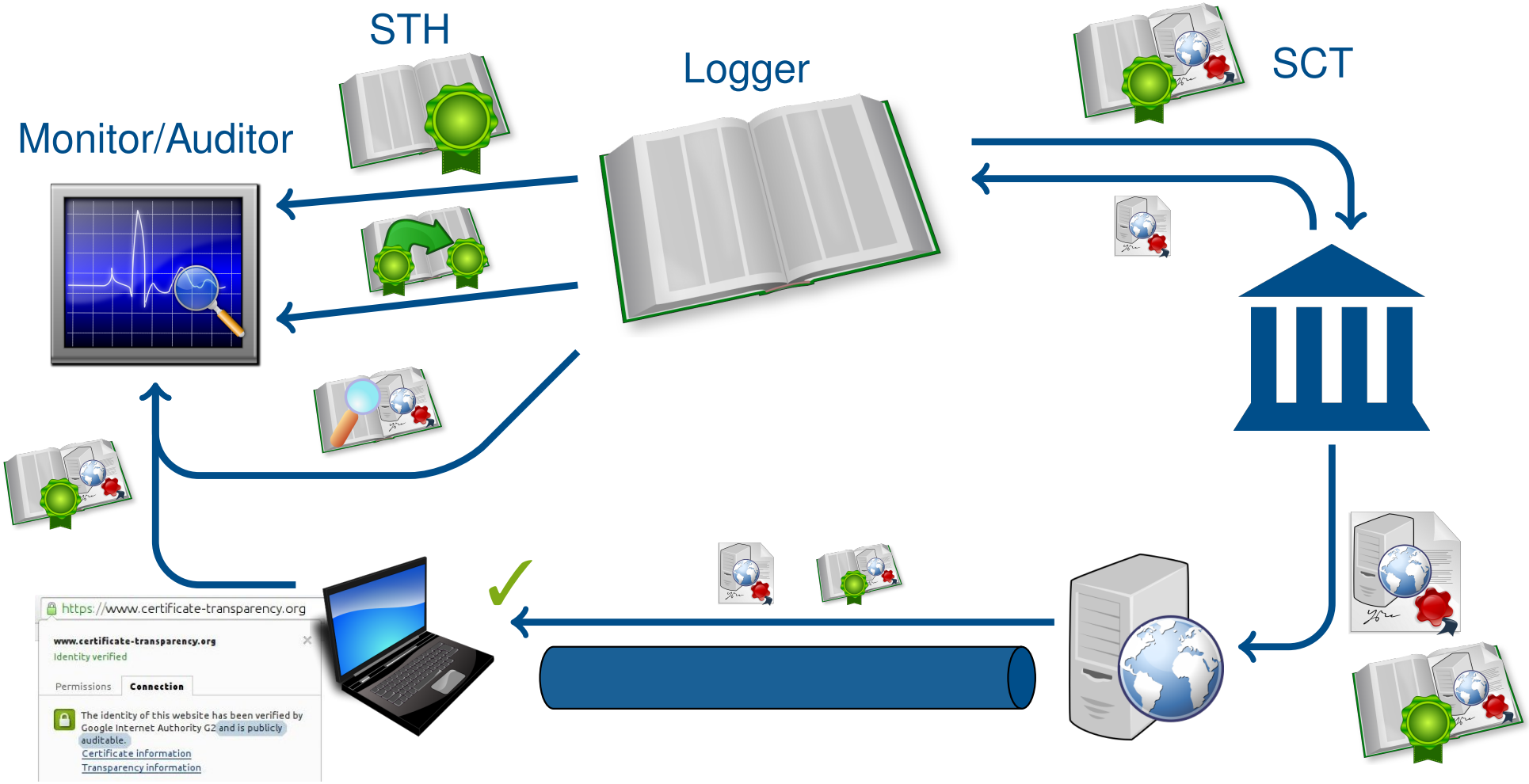
- end goal: clients only accept publicly logged certificates
- impossible for CAs to issue rogue certificate without being publicly visible

- effective
  - Sep 2015: unrequested Google certs by Thawte detected, revoked within 3 days
  - early 2016: Facebook detects certs violating policy, revoked within hours

Certificate Transparency

System Architecture

TECHNISCHE UNIVERSITÄT DARMSTADT



- multiple loggers and monitors/auditors (run by various stakeholders)

Certificate Transparency

Threat Model

TECHNISCHE UNIVERSITÄT DARMSTADT

(according to informational IETF draft)

Misbehaving Log Server

- Create entry for fake certificate
- **Present different log entry views for entities**
- Issue SCT for fake certificate
- Include syntactically invalid certificate
- ...

focus on
cryptographically assured
security properties

Misbehaving Monitor/Auditor

- **Issue false warnings about log server**
- Not inform domain owner about fake certificates
- ...

Secure Logging Schemes

TECHNISCHE UNIVERSITÄT DARMSTADT

- we introduce a cryptographic model for generic **logging schemes**
- abstracts away details of Certificate Transparency
- e.g., also allows to capture aspects of CONIKS [MBBFF @ USENIX'15]

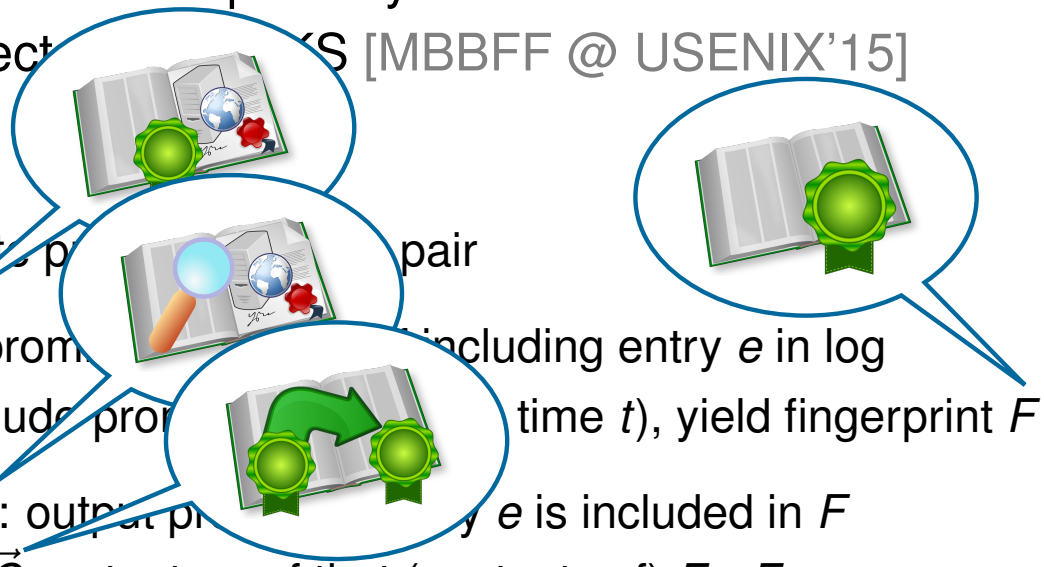
- **Logger**
  - KeyGen() $\xrightarrow{\$}$ (pk, sk): generate public/secret key pair
  - PromiseEntry$(e, t, sk) \xrightarrow{\$} P$: promise (at time t) of including entry e in log
  - UpdateLog$(\vec{P}, t, sk) \xrightarrow{\$} F$: include promised entries (at time t), yield fingerprint F
  - ProveMembership$(e, F) \xrightarrow{\$} \vec{M}$: output proof that entry e is included in F
  - ProveConsistency$(F_0, F_1) \xrightarrow{\$} \vec{C}$: output proof that (contents of) F_0, F_1 are consistent (i.e., F_0 contains prefix of entries of F_1 — "append-only")

- **Monitor/Auditor**: according Check... algorithms

Secure Logging Schemes

TECHNISCHE UNIVERSITÄT DARMSTADT

- we introduce a cryptographic model for generic **logging schemes**
- abstracts away details of Certificate Transparency
- e.g., also allows to capture aspect[illegible]S [MBBFF @ USENIX'15]

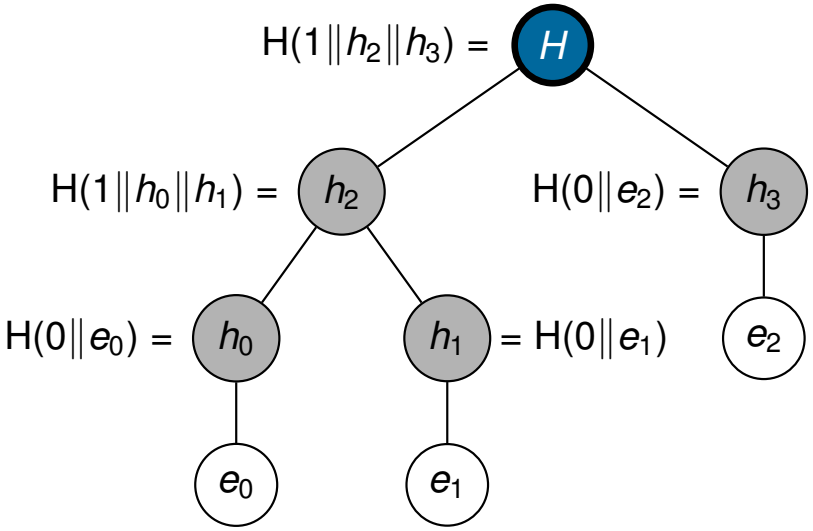


- **Logger**
  - KeyGen() $\xrightarrow{\$}$ (pk, sk): generat[illegible] p[illegible] pair
  - PromiseEntry(e, t, sk) $\xrightarrow{\$}$ P: prom[illegible] including entry e in log
  - UpdateLog$(\vec{P}, t, sk)$ $\xrightarrow{\$}$ F: includ[illegible] pro[illegible] time t), yield fingerprint F
  - ProveMembership(e, F) $\xrightarrow{\$}$ $\vec{M}$: out[illegible] e is included in F
  - ProveConsistency(F_0, F_1) $\xrightarrow{\$}$ $\vec{C}$: output proof that (contents of) F_0, F_1 are consistent (i.e., F_0 contains prefix of entries of F_1 — "append-only")
- **Monitor/Auditor**: according Check... algorithms

Certificate Transparency as a Logging Scheme (informal)

- KeyGen: generate signing keys

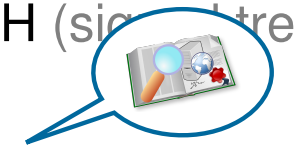
- PromiseEntry: sign entry and time as SCT (signed certificate timestamp)
- UpdateLog: add promised entries $\vec{P}.e$ to entry list $\vec{E}$,
  compute Merkle tree hash $\mathtt{MTH}(\vec{E})$ of entry list,



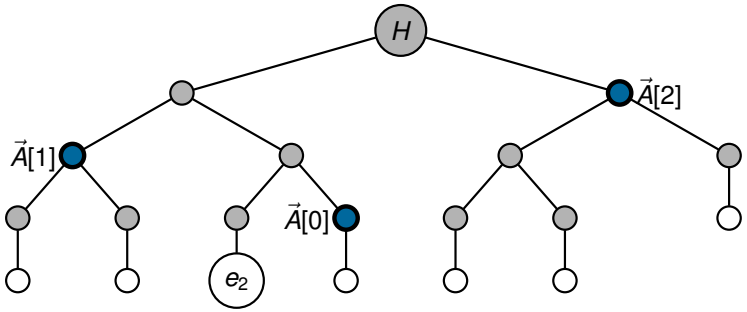
Certificate Transparency as a Logging Scheme (informal)

- KeyGen: generate signing keys

- PromiseEntry: sign entry and time as SCT (signed certificate timestamp)
- UpdateLog: add promised entries $\vec{P}.e$ to entry list $\vec{E}$,
  compute Merkle tree hash $\mathtt{MTH}(\vec{E})$ of entry list,
  output $\mathtt{MTH}(\vec{E})$ + (timed) signature as STH (signed tree head)

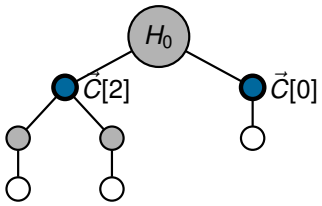


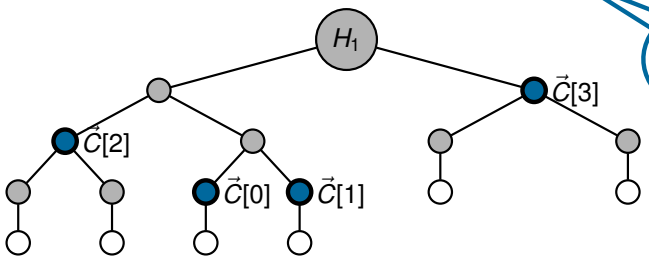
- ProveMembership: output authentication path from entry leaf to tree root STH



Certificate Transparency as a Logging Scheme (informal)

- KeyGen: generate signing keys
- PromiseEntry: sign entry and time as SCT (signed certificate timestamp)
- UpdateLog: add promised entries $\vec{P}.e$ to entry list $\vec{E}$,
  compute Merkle tree hash $\mathtt{MTH}(\vec{E})$ of entry list,
  output $\mathtt{MTH}(\vec{E})$ + (timed) signature as STH (signed tree head)
- ProveMembership: output authentication path from entry leaf to tree root STH
- ProveConsistency: output Merkle tree consistency proof between STHs





Cryptographic Security Goals

TECHNISCHE UNIVERSITÄT DARMSTADT

Security Against a Malicious Logger

- Collision resistance of entries
  hard to create fingerprint F representing two different entry sets $\vec{E}_0$, $\vec{E}_1$
- Collision resistance of proofs
  hard to create proof that entry e is in F and also have F represent $\vec{E} \not\ni e$
- Consistency of entries
  hard to create consistency proof for F_0, F_1 while $\vec{E}_0$ isn't a prefix of $\vec{E}_1$

Security Against a Malicious Monitor/Auditor

- Inclusion of promises
  hard to frame an honest logger for not including promised entries

Cryptographic Security Goals

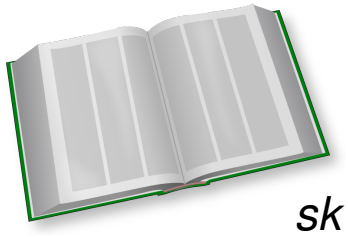
Example: Inclusion of Promises

TECHNISCHE UNIVERSITÄT DARMSTADT

(Malicious Monitor/Auditor)

(Honest Logger)

pk



sk

OPromiseEntry( )

OUpdateLog(, , . . .)

OProveMembership(,)

OProveConsistency(,)

OTick() , include pending promised entries

promised not included (in time) in fingerprint

Security of Certificate Transparency

TECHNISCHE UNIVERSITÄT DARMSTADT

Security Against a Malicious Logger

- ✓ Collision resistance of entries
- ✓ Collision resistance of proofs
- ✓ Consistency of entries

given hash function is **collision-resistant**

Security Against a Malicious Monitor/Auditor

- ✓ Inclusion of promises

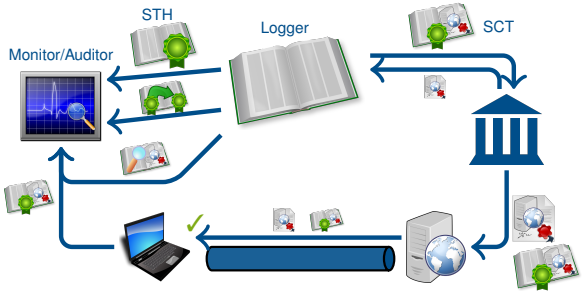
given hash function is **collision-resistant**
and signature scheme is **(existentially) unforgeable**

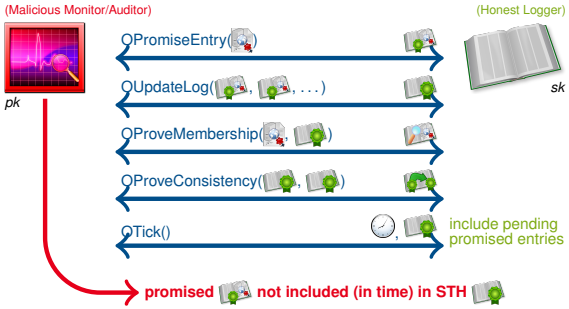
Summary

TECHNISCHE UNIVERSITÄT DARMSTADT

We

- propose a cryptographic model for secure logging schemes
- formalize game-based security notions against malicious loggers and monitors/auditors
- establish security of Certificate Transparency in our model under reasonable assumptions
- discuss generality of our model and applications to related settings





full version @ IACR ePrint or `www.felixguenther.info`

- `https://ia.cr/2016/452`

Thank You!