TU Graz

SCIENCE
PASSION
TECHNOLOGY

Revisiting User Privacy for Certificate Transparency

Daniel Kales, Olamide Omolola, Sebastian Ramacher

EuroS&P'19, June 19th, 2019

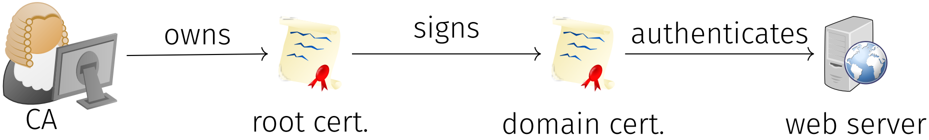
Supported by: iOV42 LIGHTest

Outline

What is Certificate Transparency?

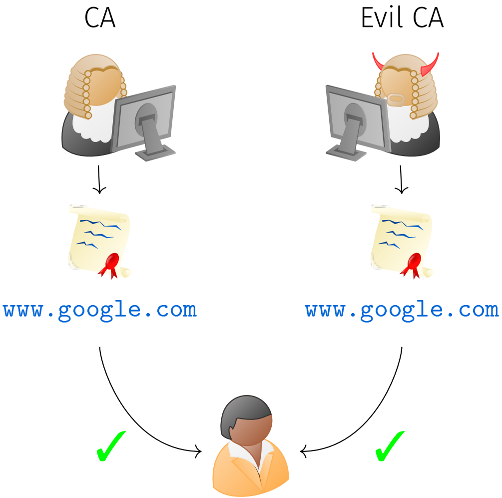
X509 Certificate Ecosystem

- Certificates bind user's information to the public key
- Certificate is signed by a root certificate
- Root certificate is owned by a trusted entity called Certificate Authority (CA)
- User's certificate can be verified by linking it to the known root certificate



X509 Certificate Ecosystem Problems

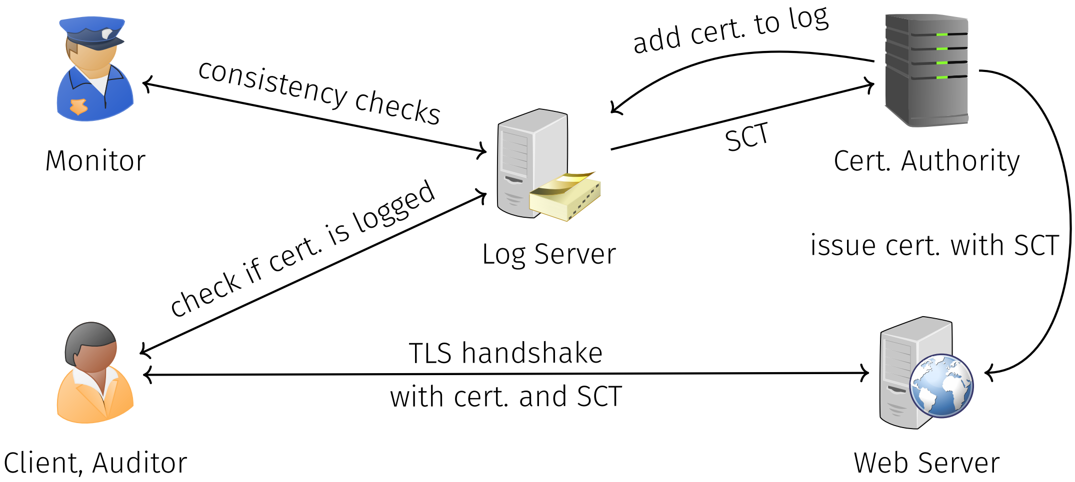
- Signatures prevent malicious websites from using forged certificates
- No protection against mistakenly or maliciously issued certificates!
- Real-world problems: DigiNotar compromised by hackers



Certificate Transparency

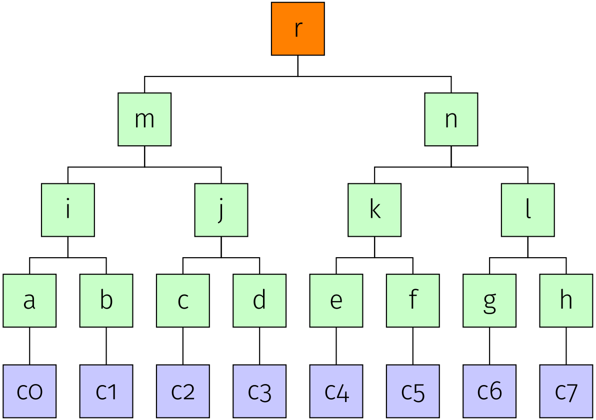
- Need monitoring system for all issued certificates
- Goals:
  - easily accessible to everyone, open framework
  - refuse use of certificates not in monitoring system
  - cryptographic guarantees for logging
- **Certificate Transparency** [Lau14; LLK13] was designed to be this system
- Log servers give signed promise of inclusion in log to CA
  - Signed Certificate Timestamp (SCT)
- **Mandatory for certificates issued after April 30th, 2018!**

Certificate Transparency (cont.)



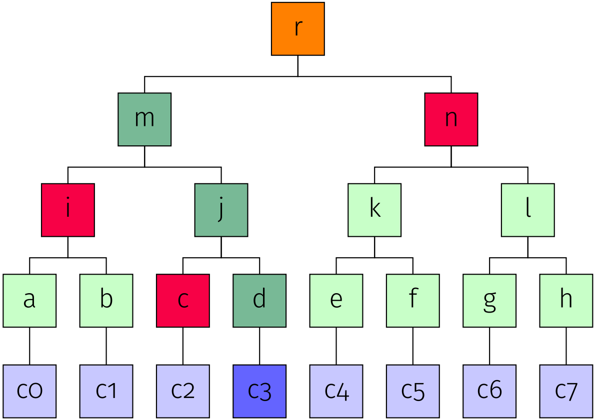
Log Server Structure

- Merkle tree
  - Binary tree of hashed nodes
  - Log server periodically updates tree with new certificates
  - Log server also signs root hash



Log Server Structure (cont.)

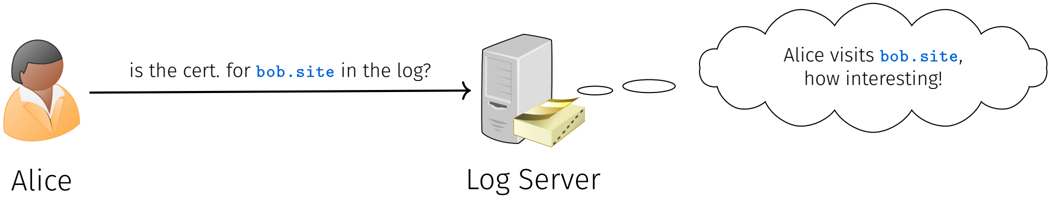
- Membership proof
  - Release intermediary hashes
  - Re-calculate path to root
  - Compare against known root hash
  - Logarithmic proof size



Privacy Concerns for End Users

Privacy Concerns for End Users

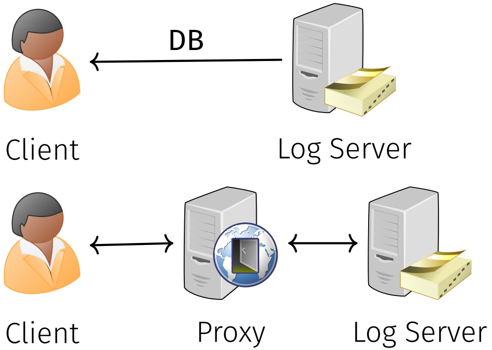
- End users have auditing role
  - Verify certificate is contained in log according to SCT
  - If not, report log server as malicious
- Privacy loss:
  - Log server learns browsing behavior of client
  - Could deter clients from using Certificate Transparency



Solving Privacy Concerns

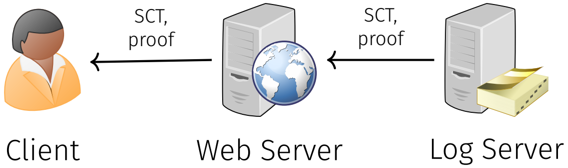
Naïve Solutions

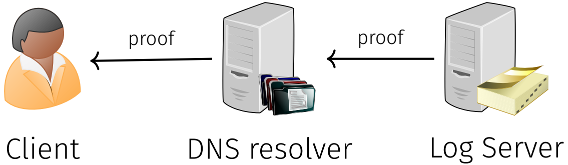
- Download full log:
  - Infeasible for most clients
  - Log sizes of 10+ GiB
- Redirect query through proxy:
  - Protect client query from log server
  - Only shifts privacy problem to proxy



Other Approaches

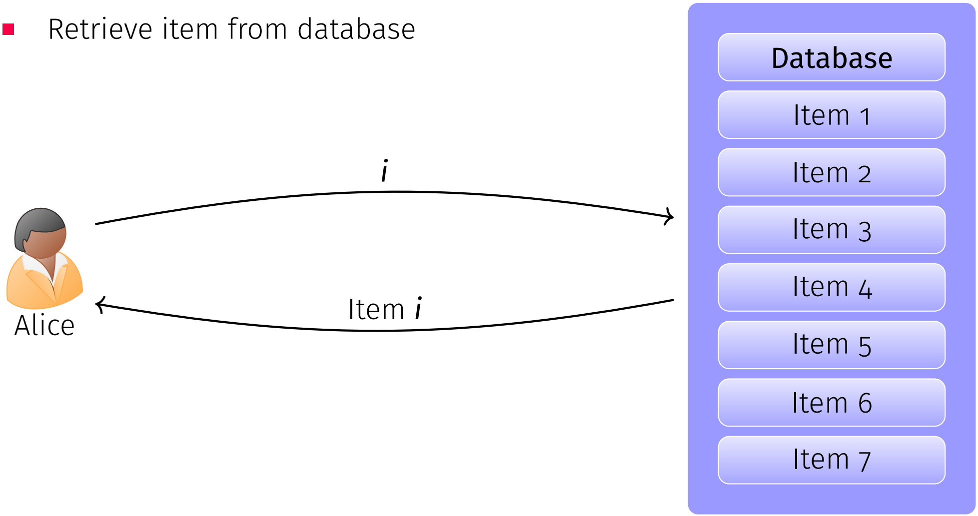
- Stapling Approach:
  - Web server gets proof from log server
  - Forwards proof to client
  - More work for web server
- CT over DNS:
  - Get proofs via DNS queries
  - Shifts privacy concerns to DNS server
  - DNS mostly plaintext





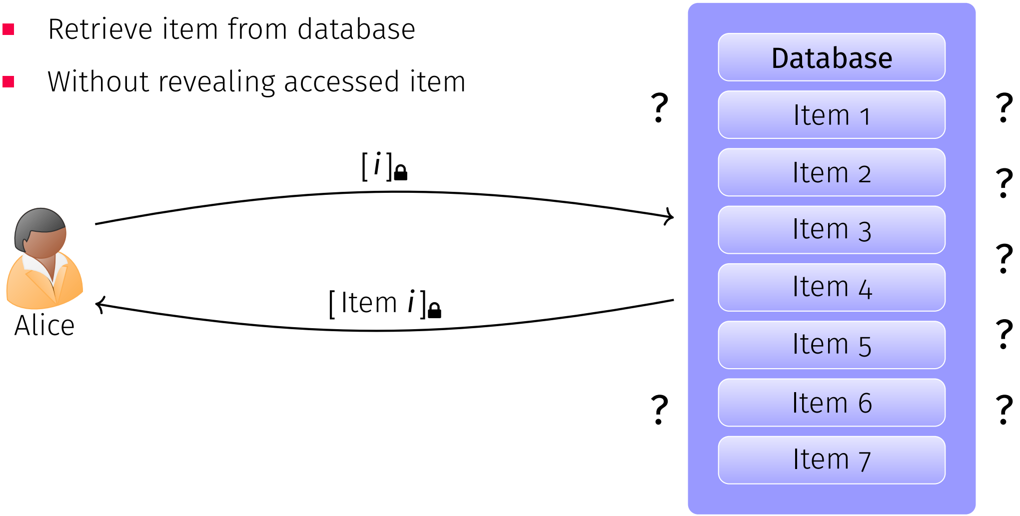
Private Information Retrieval

- Retrieve item from database



Private Information Retrieval

- Retrieve item from database
- Without revealing accessed item



Private Information Retrieval (cont.)

- Previous efforts by Lueks and Goldberg [LG15] in 2015
  - Optimizations to Percy++ PIR system
  - Multi-server model
  - Speedup when answering many client queries at once
  - Use-case: Certificate Transparency
- Assumed 4 million certificates
  - Runtime of a few seconds per query
  - Practical today?

Current Log Server Statistics

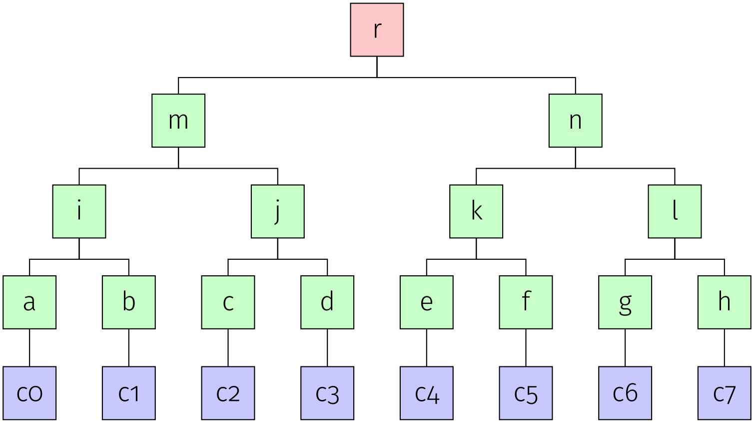
- `merkle.town`: CT ecosystem statistics[1]
- Number of new certificates per hour (global): $\approx 53,000$

Root CA	Certificates		Percentage
DigiCert	64,226,041	($2^{25.94}$)	5%
Let's Encrypt	941,016,262	($2^{29.81}$)	72%
Sectigo	246,484,842	($2^{27.88}$)	19%
Other	62,114,615	($2^{25.89}$)	5%

[1] retrieved on 2019-04-08

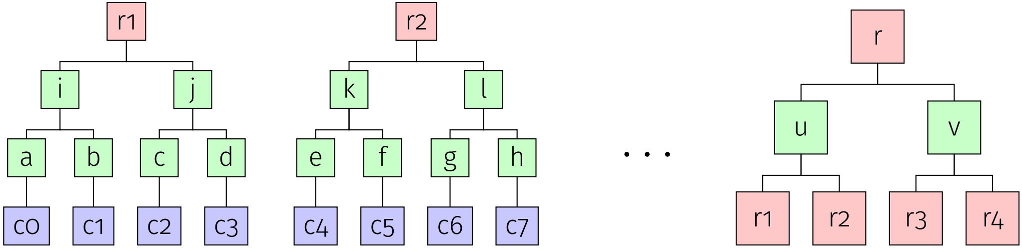
Changes to Merkle Tree Structure

- Original Merkle Tree:



Changes to Merkle Tree Structure (cont.)

- Merkle Tree with Sub-Trees:



Changes to Merkle Tree Structure (cont.)

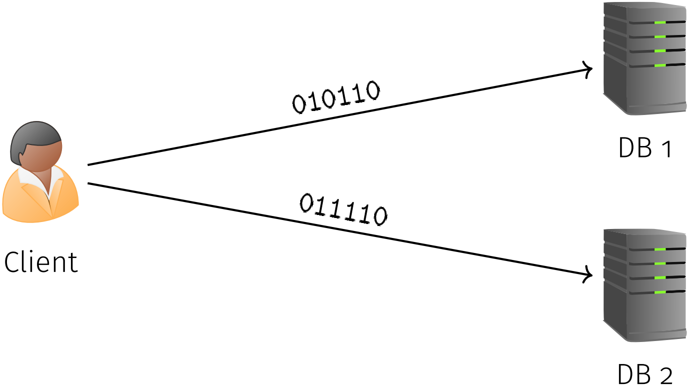
- Log server only **issues** SCTs **once per** predetermined **time span**
  - e.g., a new sub-tree every hour (time span configurable)
- By completing full sub-tree, we can **include the proof** in the SCT
  - store Merkle tree proofs for sub-trees in SCT **extension** field
  - only retrieve proof between sub-tree root and top-level root hashes
- Tradeoff between tree size, SCT issuing latency and SCT size
- Other accumulators possible (e.g., bilinear accumulators)

Multi-Server PIR

- Multi-Server PIR gives information-theoretic security
  - Even with unlimited computing power, no way for server to find index i!
- Important restriction: **No collusion** between servers!
- Much better performance than single-server PIR
  - No need for expensive primitives, e.g. homomorphic encryption
  - Based on secret-sharing approaches

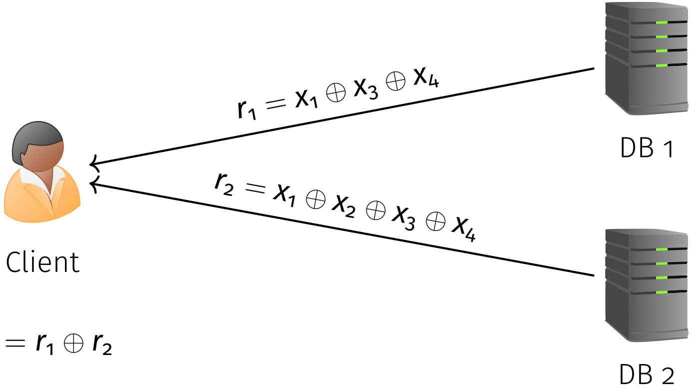
Multi-Server PIR (cont.)

- "Linear-Summation Scheme" [CGK+95]



Multi-Server PIR (cont.)

- "Linear-Summation Scheme" [CGK+95]

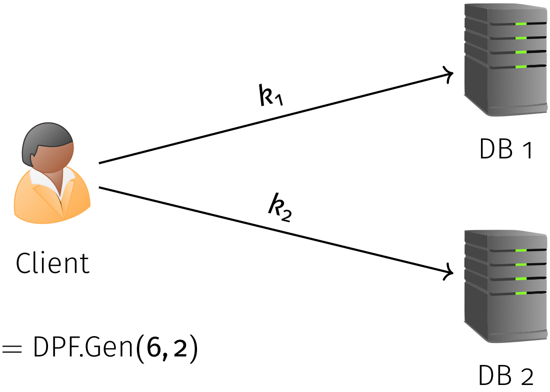


$$x_2 = r_1 \oplus r_2$$

Two-Server PIR from DPFs

- Problem: Still $N = |DB|$ bits of communication per server and query
- Distributed Point Functions (DPF) [GI14]
  - "Function Secret Sharing" by Boyle et al. [BGI15]
- $(k_1, k_2) \leftarrow \text{DPF.Gen}(N, q)$
  - Generate two short ($\log N$) keys based on chosen index q and length N
- $K_i \leftarrow \text{DPF.Eval}(N, k_i)$
  - Expand short key k_i to N bit long keystream K_i
- Property: $K_1 \oplus K_2$ is a bitstring with only one bit at position q set

Two-Server PIR from DPFs (cont.)



$010110 = \text{DPF.Eval}(6, k_1)$

$011110 = \text{DPF.Eval}(6, k_2)$

$(k_1, k_2) = \text{DPF.Gen}(6, 2)$

Multi-Server PIR Deployment

- Important requirement: **No collusion between two servers!**
  - If violated, privacy is lost!
- Real-world deployment:
  - Log server data is publicly accessible
  - Competitor of first log server: Google $\leftrightarrow$ Microsoft
  - Privacy-conscious organizations: EFF, EDRi
- Only extension to normal log server API, users still can query without privacy protection

Practical Evaluation

Evaluation

- Single-Server PIR
  - Not feasible for full CT logs with 2^{28} or more elements!
  - Open-source PIR framework XPIR [MBF+16]
  - Evaluation for sub-tree each hour

tree size	DB gen.	Query gen. [ms]	Reply gen.	Comm. [KB]
2^{15}	3640	7491	1748	128

Evaluation (cont.)

- Multi-Server PIR using DPFs
  - Almost feasible even for full CT logs!

tree size	DPF.Gen	DPF.Eval [ms]	XOR	Total	Comm. [B]
2^{20}	0.05	0.32	4.28	4.66	2938
2^{22}	0.07	1.23	16.72	18.03	3590
2^{24}	0.08	4.78	64.49	69.36	4314
2^{26}	0.09	19.22	251.32	270.64	5110
2^{28}	0.11	78.41	988.93	1067.46	5978

Evaluation (cont.)

- Multi-Server PIR using DPFs and sub-accumulators
  - Overhead less than 10 ms and 4 KB for full CT log

N	N_Λ	Sub-acc. type	N_{sub}	DPF total [ms]	Acc. verify [ms]	Com. [B]	extra [B]
2^{31}	2^{15}	RSA	2^{16}	0.13	3.97	1623	384
2^{31}	2^{15}	Bilinear	2^{16}		2.81		768
2^{31}	2^{15}	Merkle	2^{16}		< 0.01		512
2^{31}	2^{21}	RSA	2^{10}	8.36	3.97	3255	384
2^{31}	2^{21}	Bilinear	2^{10}		2.81		768
2^{31}	2^{21}	Merkle	2^{10}		< 0.01		320

Conclusion

- Changes to Merkle-Tree structure enable less costly PIR queries
- Sub-tree structure generalizes to other types of accumulators
- Multi-server PIR based on DPFs with sub-accumulators
  - Overhead less than 10 ms and 4 KB for full CT log
- Multi-server PIR possible without major changes in CT ecosystem
- Optional for users if they want privacy, compatible with old API

Questions?

Implementation:

- DPF (in C++): https://github.com/dkales/dpf-cpp
- DPF (in Go): https://github.com/dkales/dpf-go
- Log server: https://github.com/dkales/certificate-transparency

References I

[BGI15] Elette Boyle, Niv Gilboa, and Yuval Ishai. **Function secret sharing**. EUROCRYPT (2), volume 9057 of *Lecture Notes in Computer Science*, pages 337–367. Springer, 2015.

[CGK+95] Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. **Private information retrieval**. FOCS, pages 41–50. IEEE Computer Society, 1995.

[GI14] Niv Gilboa and Yuval Ishai. **Distributed point functions and their applications**. EUROCRYPT, volume 8441 of *Lecture Notes in Computer Science*, pages 640–658. Springer, 2014.

[Lau14] Ben Laurie. **Certificate transparency**. *ACM Queue*, 12(8):10–19, 2014.

[LG15] Wouter Lueks and Ian Goldberg. **Sublinear scaling for multi-client private information retrieval**. Financial Cryptography, volume 8975 of *Lecture Notes in Computer Science*, pages 168–186. Springer, 2015.

[LLK13] Ben Laurie, Adam Langley, and Emilia Käsper. **Certificate transparency**. *RFC*, 6962:1–27, 2013.

[MBF+16] Carlos Aguilar Melchor, Joris Barrier, Laurent Fousse, and Marc-Olivier Killijian. **XPIR : private information retrieval for everyone**. *PoPETs*, 2016(2):155–174, 2016.