

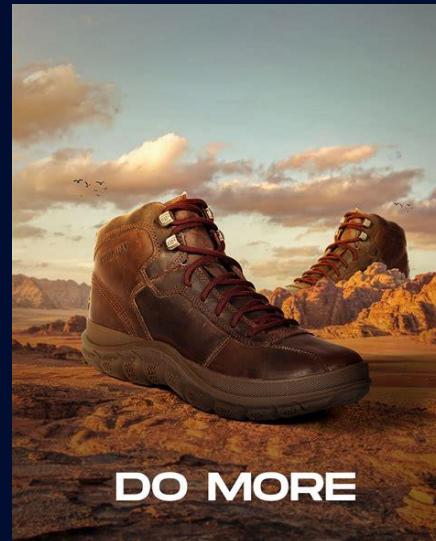
# Bandwidth Efficient Partial Authorized PSI

Tjitske Koster, Francesca Falzon, Lilika Markatou

# Advertisements



We want to  
advertise  
shoes...



We'll sell  
you adds!



# Advertisement strategy



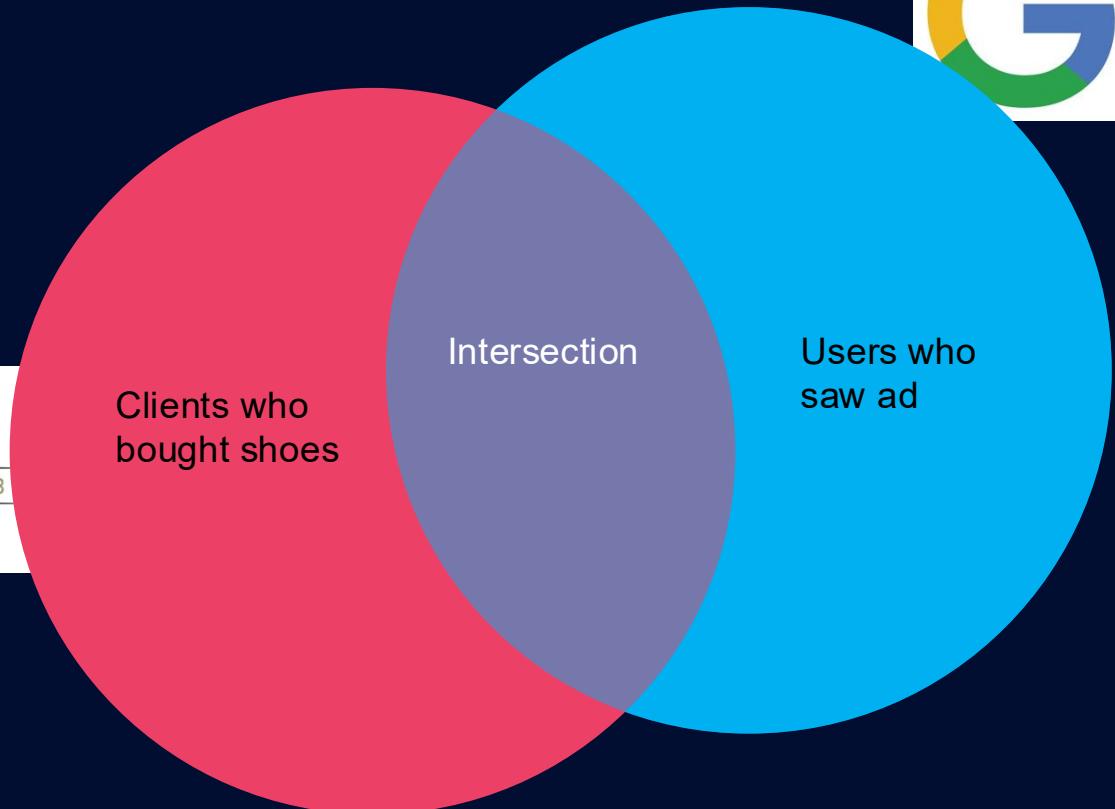
We sold  
shoes, but  
did they see  
the ad?

What if...

If you send  
your clients,  
I'll tell you!



# Private set intersection





All  
internet  
users



Intersection  
– All users  
who saw  
the add

# But...

Many attacks exist



Willem-  
Alexander



Users  
who  
saw  
ad



**Birds of a Feather Flock Together:  
How Set Bias Helps to Deanonymize You  
via Revealed Intersection Sizes**

Xiaojie Guo, Ye Han, Zheli Liu, Ding Wang, and  
Yan Jia, Nankai University; Jin Li, Guangzhou University  
<https://www.usenix.org/conference/usenixsecurity22/presentation/guo>



## Learning from Functionality Outputs: Private Join and Compute in the Real World

Francesca Falzon  
ETH Zürich, Switzerland

Tianxin Tang  
Eindhoven University of Technology, Netherlands

### Abstract

Private Join and Compute (PJC) is a two-party protocol recently proposed by Google for various use-cases, including ad conversion (AsiaCrypt 2021) and which generalizes their deployed private set intersection sum (PSI-SUM) protocol (EUROCRYPT 2020). PJC allows two parties to compute the intersection sum of their sets while maintaining privacy.

Set Intersection Sum (PSI-SUM) protocol warns that the intersection-sum *could* reveal something about the intersection [19, 20]. Their suggestions for mitigating such attacks include scrubbing inputs to remove “outliers”, aborting if the intersection-size is too small, and adding noise to the output.

But...

Many attacks exist

## On the Insecurity of Bloom Filter-Based Private Set Intersections

Jelle Vos , Delft University of Technology

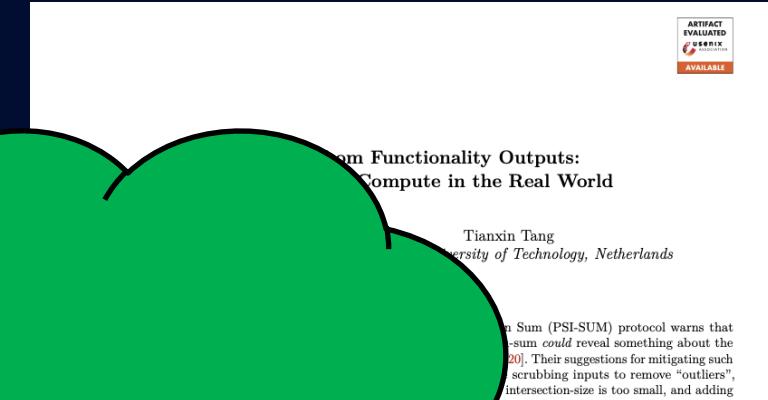
Jorrit van Assen , Delft University of Technology

Tjitske Koster , Delft University of Technology

Evangelia Anna Markatou , Delft University of Technology



ABSTRACT  
EVALUATED  
USENIX  
CONFERENCE  
AVAILABLE



# Introduce a Judge!

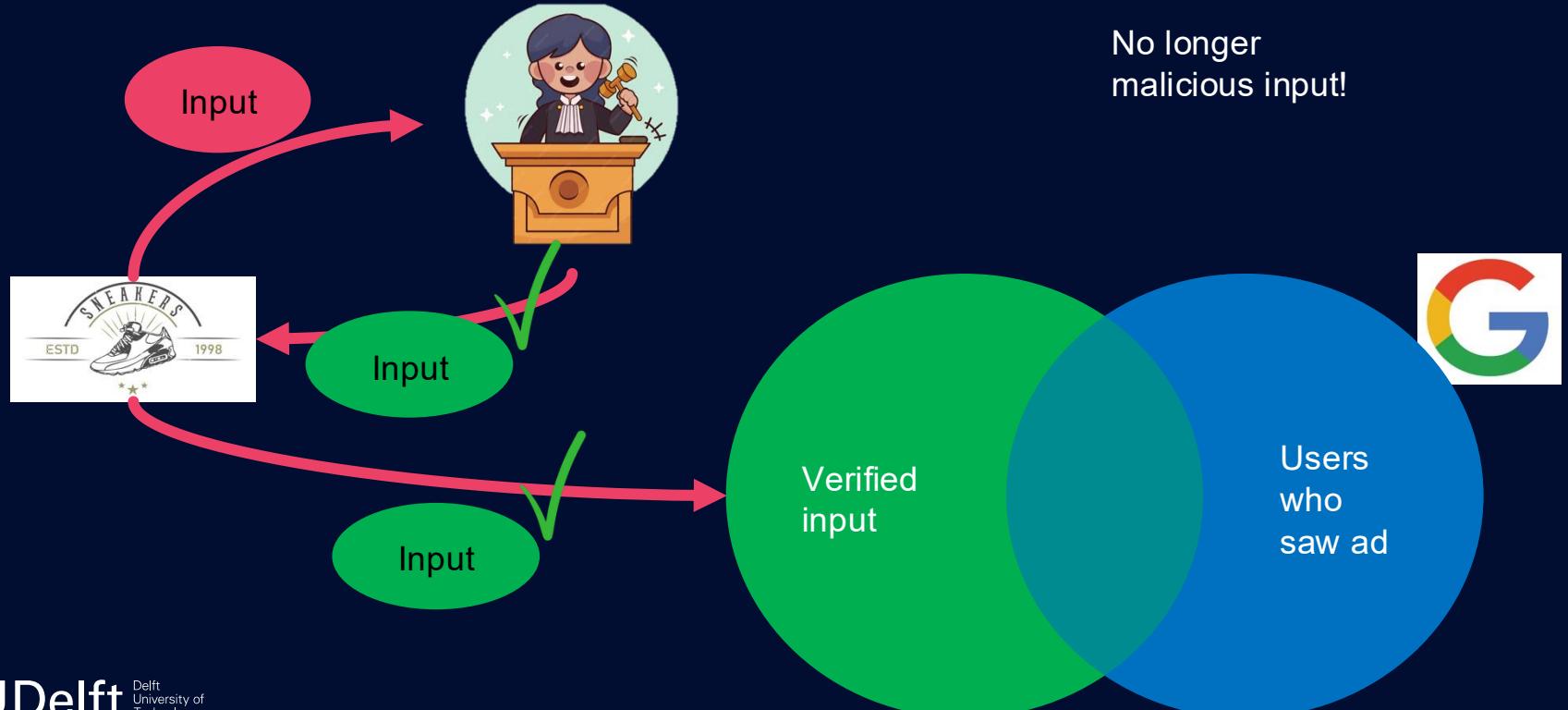
From Filter-Based  
Functions

Tuomas J. Helonen , Delft University of Technology  
Jeroen van den Bos , Delft University of Technology  
Hans Koster , Delft University of Technology  
Evangelia Anna Markatou , Delft University of Technology

# Today

- (Partial) Authorized PSI
- Faster Partial Authorized PSI
- How partial is partial? → Game theory

# Authorized Private Set Intersection



But....

We have to fully trust the judge....

# This problem was introduced and solved by:

## **Re-visiting Authorized Private Set Intersection: A New Privacy-Preserving Variant and Two Protocols**

Francesca Falzon

[ffalzon@ethz.ch](mailto:ffalzon@ethz.ch)

ETH Zürich

Zürich, Switzerland

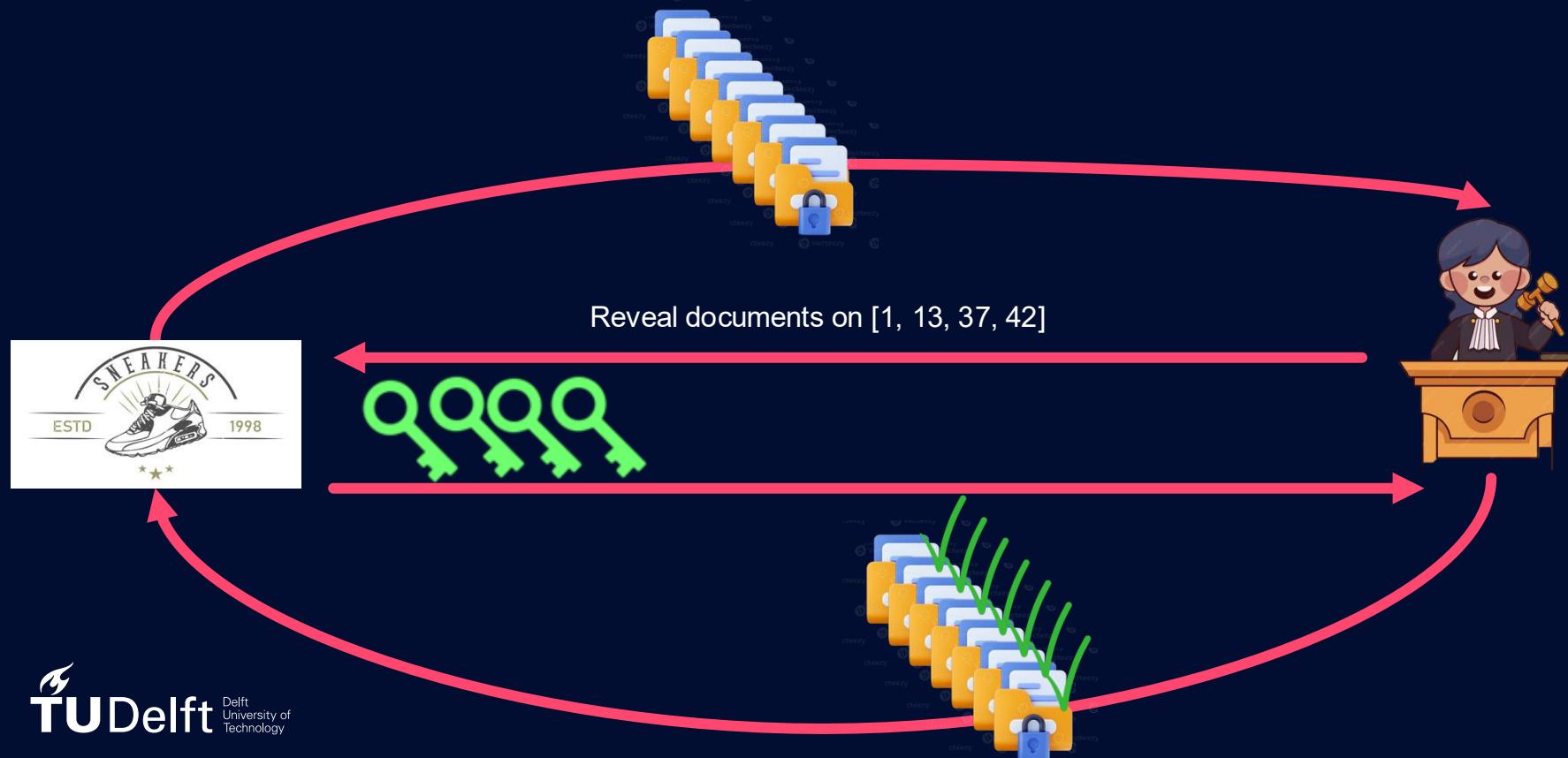
Evangelia Anna Markatou

[e.a.markatou@tudelft.nl](mailto:e.a.markatou@tudelft.nl)

TU Delft

Delft, Netherlands

# Partial Authorized Private Set Intersection



# But....

That is a lot of bandwidth

# Solution 2

Paper 2025/2132

## Bandwidth Efficient Partial Authorized PSI

*Tjitske Ollie Koster* , Delft University of Technology

*Francesca Falzon* , ETH Zurich

*Evangelia Anna Markatou* , Delft University of Technology

## Solution 2 - improvement



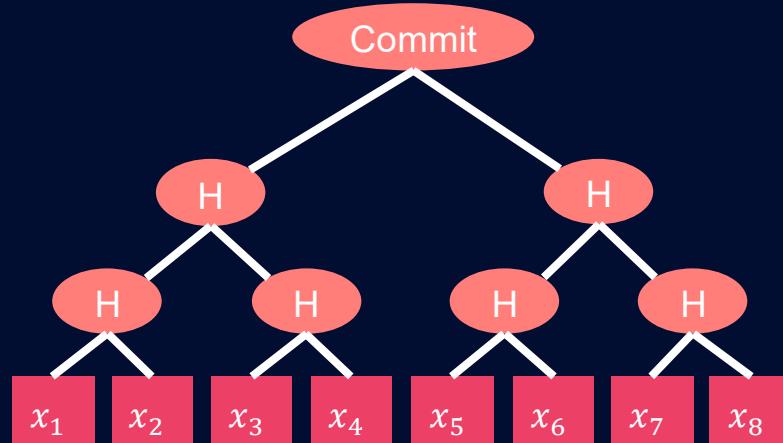
# And then intersect a box?

What is a box?

# Vector commitment

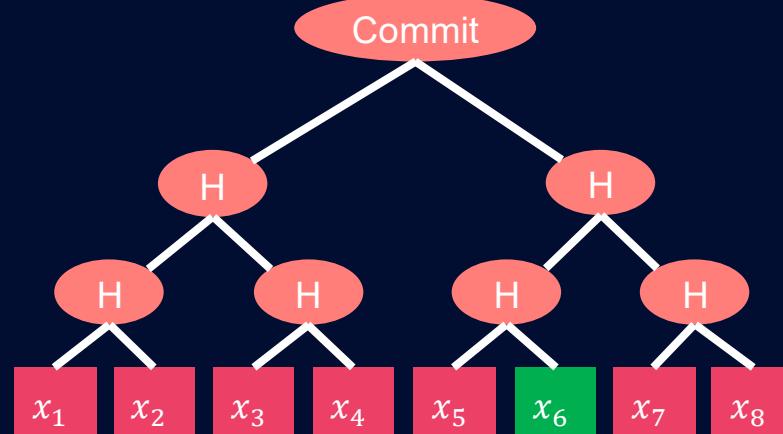
- Commit to values

Commit



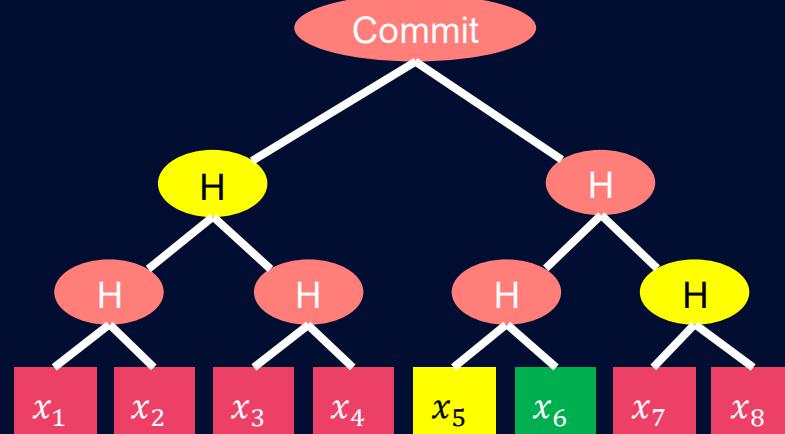
# Vector commitment

- Commit to values
- Prove value  $x_i$  on position  $i$  of vector



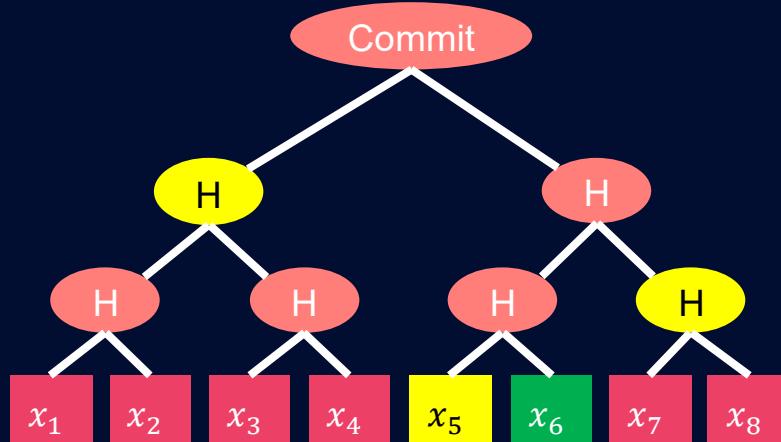
# Vector commitment

- Commit to values
- Prove value  $x_i$  on position  $i$  of vector



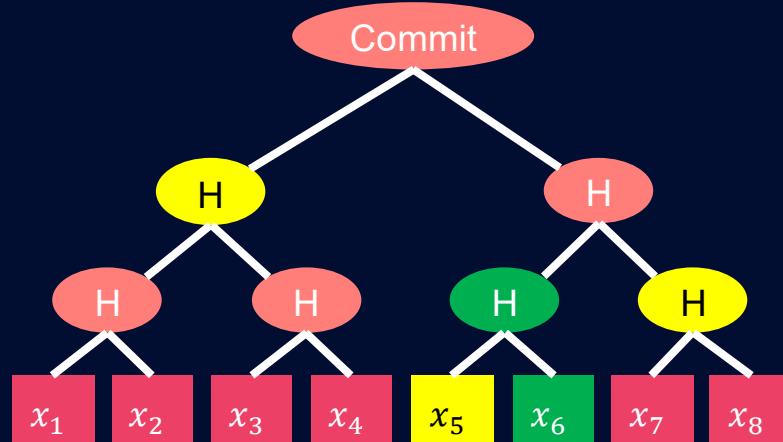
# Vector commitment

- Commit to values  
- Prove value  $x_i$  on position  $i$  of vector 
- Verify, the proof  wrt 



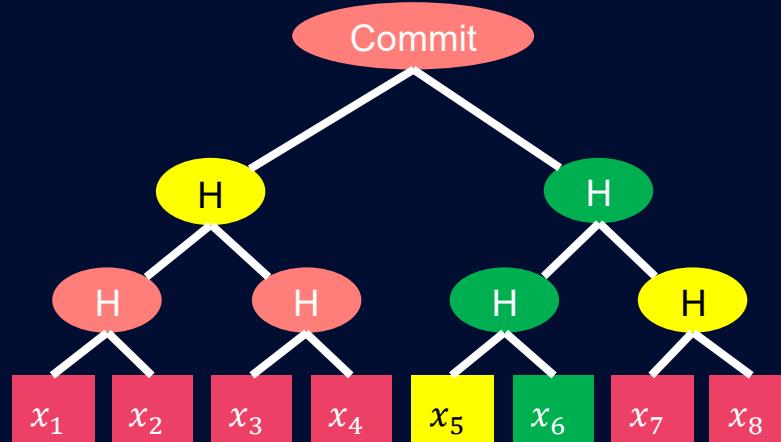
# Vector commitment

- Commit to values      Commit
- Prove value  $x_i$  on position  $i$  of vector       $\pi$
- Verify, the proof  $\pi$  wrt      Commit



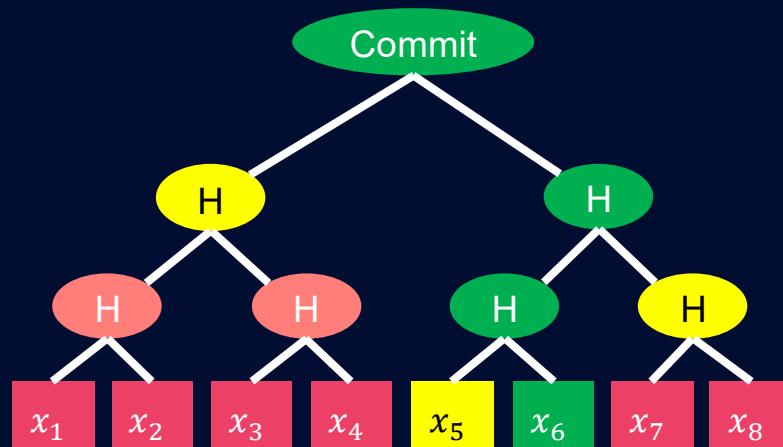
# Vector commitment

- Commit to values      Commit
- Prove value  $x_i$  on position  $i$  of vector       $\pi$
- Verify, the proof  $\pi$  wrt      Commit



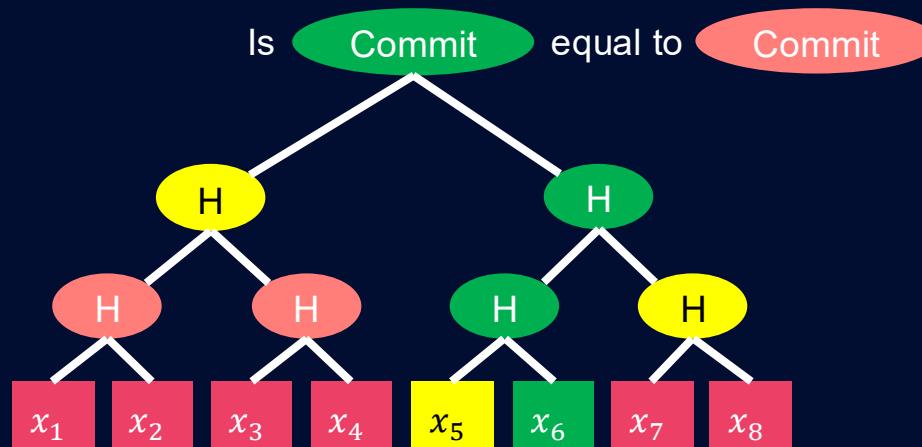
# Vector commitment

- Commit to values      Commit
- Prove value  $x_i$  on position  $i$  of vector       $\pi$
- Verify, the proof  $\pi$  wrt      Commit



# Vector commitment

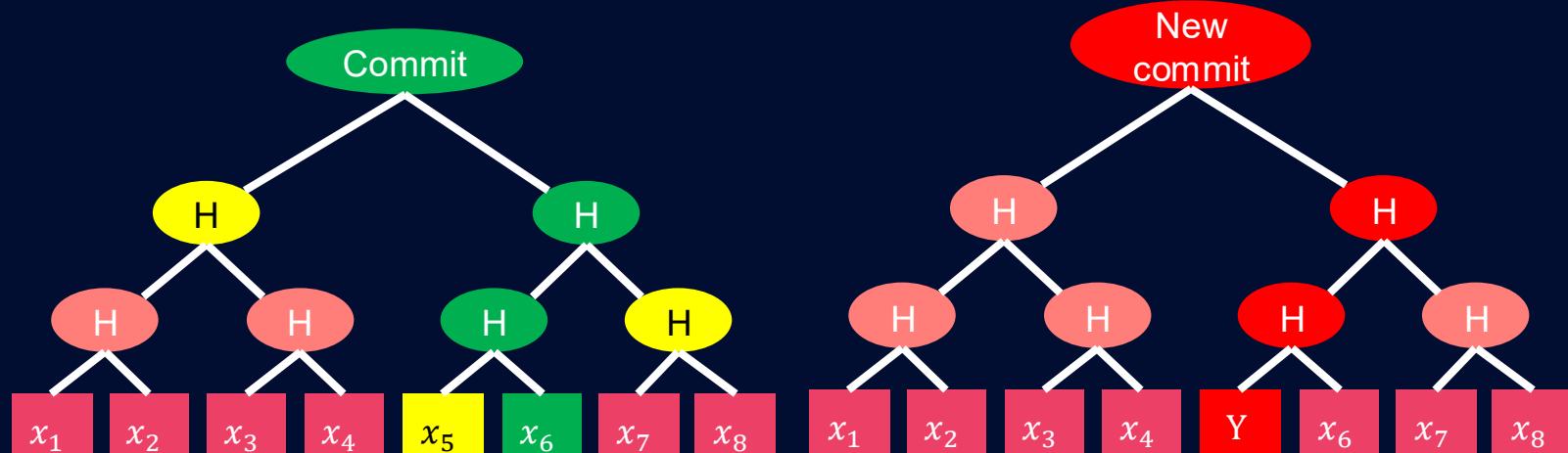
- Commit to values      Commit
- Prove value  $x_i$  on position  $i$  of vector       $\pi$
- Verify, the proof  $\pi$  wrt      Commit



# Vector commitment

- Commit to values 
- Prove value  $x_i$  on position  $i$  of vector 
- Verify, the proof  wrt 

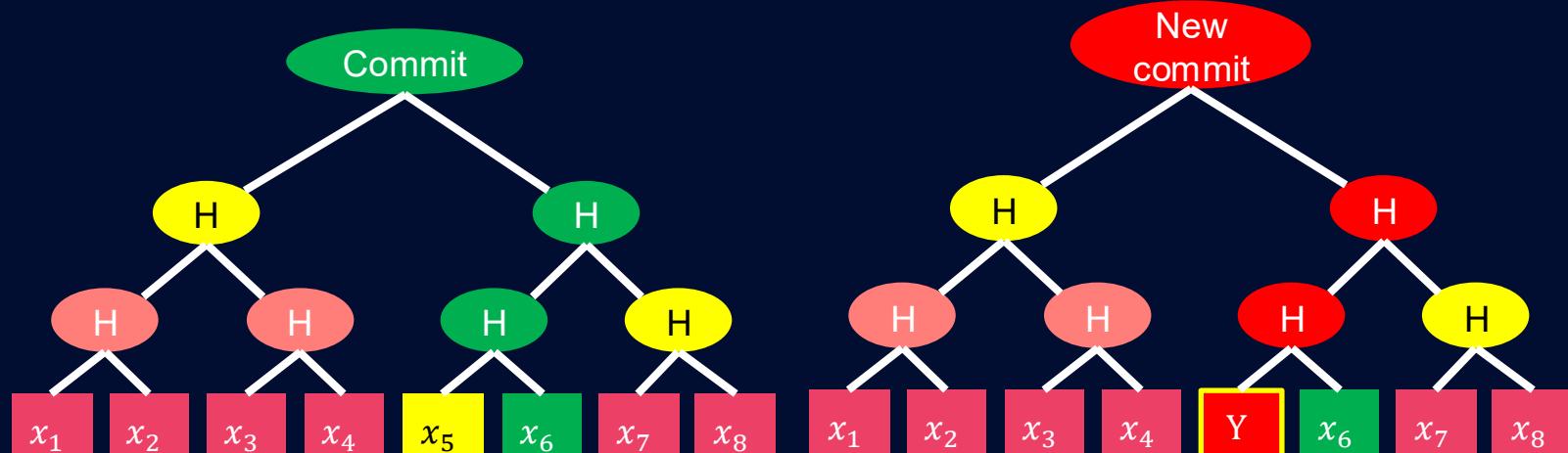
- Correctness
- No cheating



# Vector commitment

- Commit to values 
- Prove value  $x_i$  on position  $i$  of vector 
- Verify, the proof  wrt 

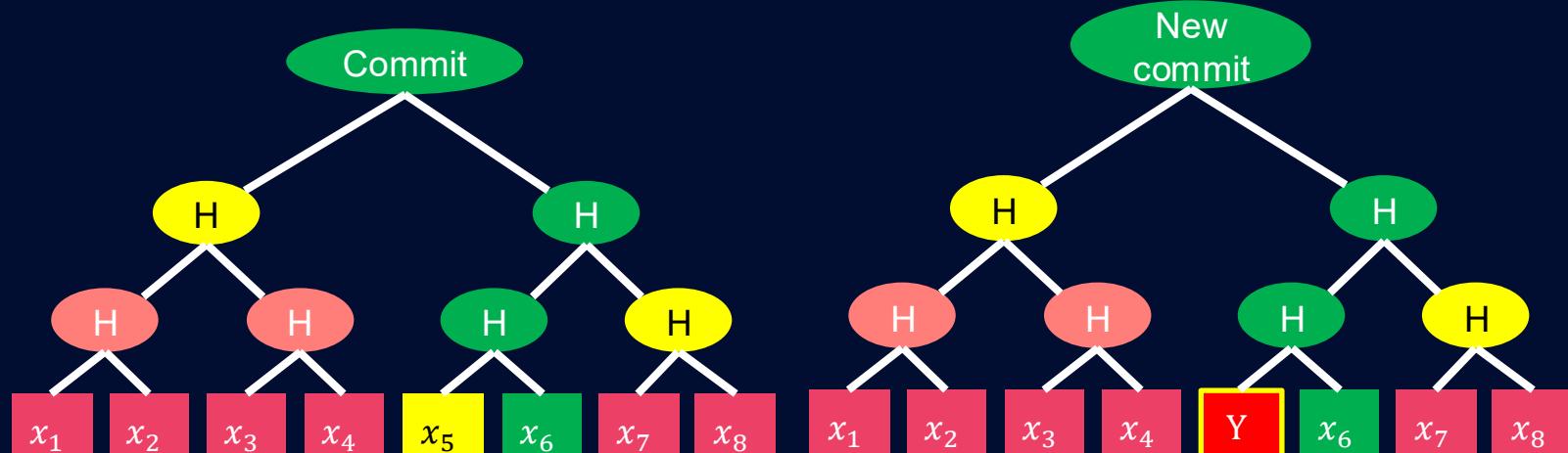
- Correctness
- No cheating



# Vector commitment

- Commit to values 
- Prove value  $x_i$  on position  $i$  of vector 
- Verify, the proof  wrt 

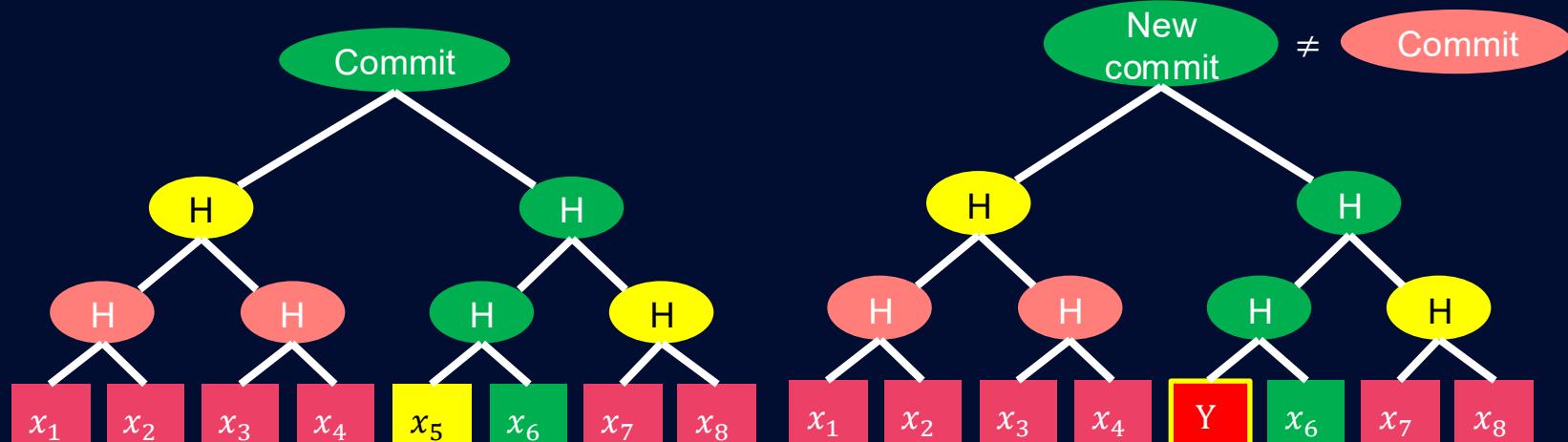
- Correctness
- No cheating



# Vector commitment

- Commit to values 
- Prove value  $x_i$  on position  $i$  of vector 
- Verify, the proof  wrt 

- Correctness
- No cheating



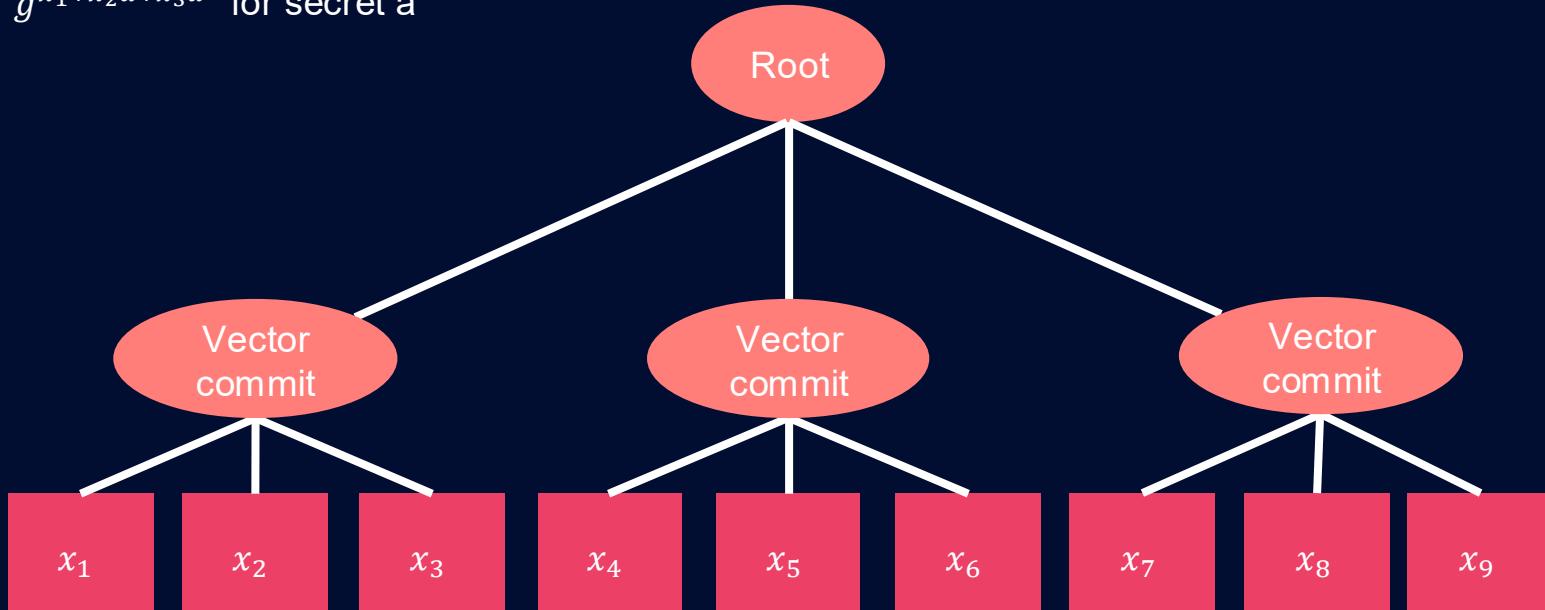
# Still a lot of bandwidth

Proof is of length  $\log_2(n)$

# Verkle tree

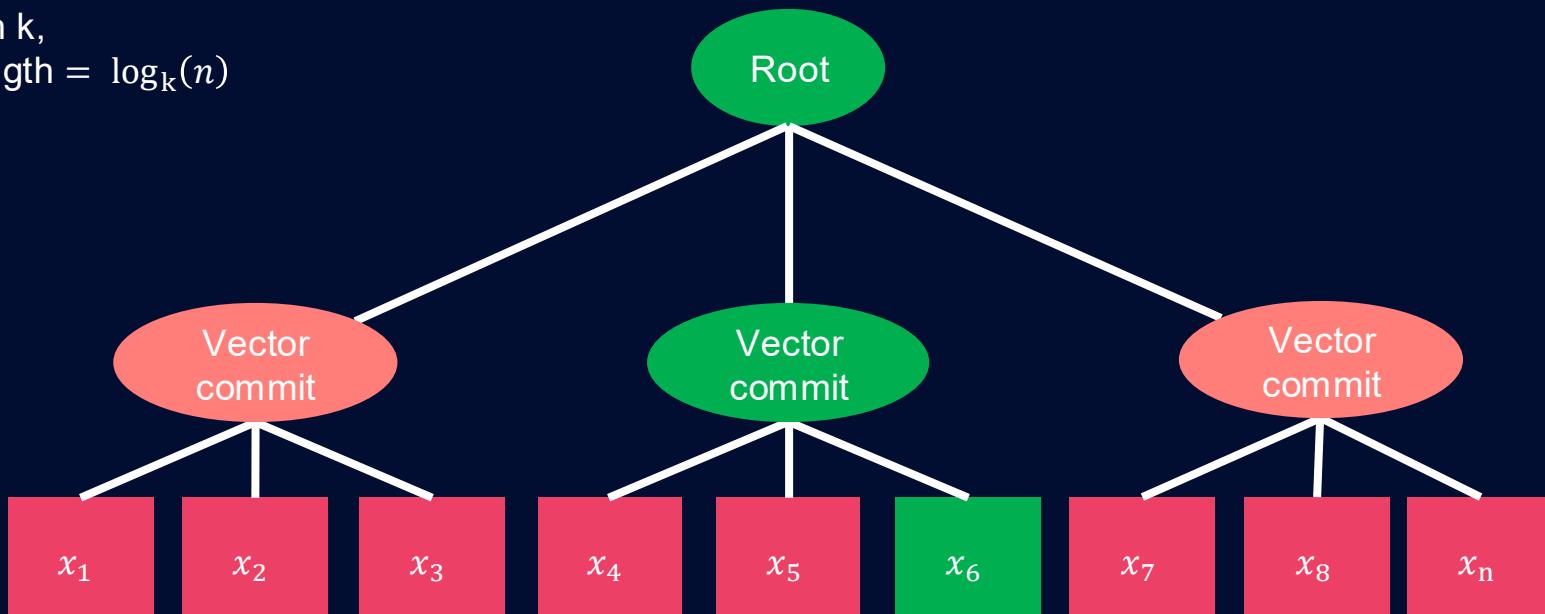
Vector commitment → Pointproofs

Commitment:  $g^{x_1+x_2a+x_3a^2}$  for secret a

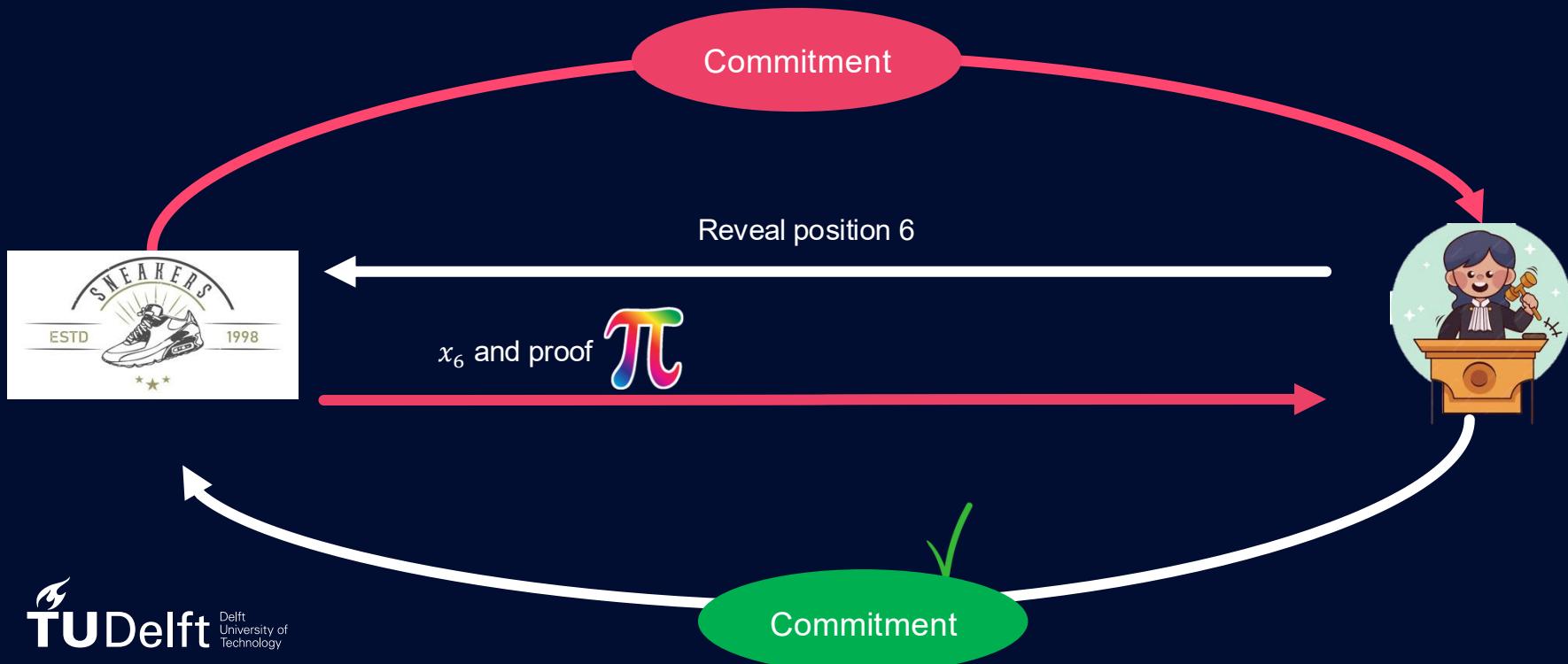


# Verkle tree - proof

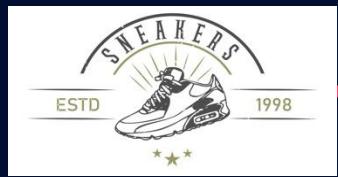
For width  $k$ ,  
Proof length =  $\log_k(n)$



# Authorization



# Intersection



Commitment

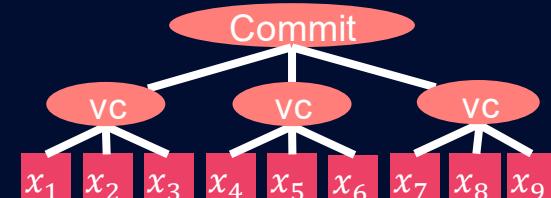


$x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7 \ x_8 \ x_9$

1. Build tree
2. Check commitment

Intersect

$y_1 \ y_2 \ y_3 \ y_4 \ y_5 \ y_6 \ y_7 \ y_8 \ y_9$



# Authorization – math version



$\{m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, m_9\}$

$$x_i = H(m_i)^r$$

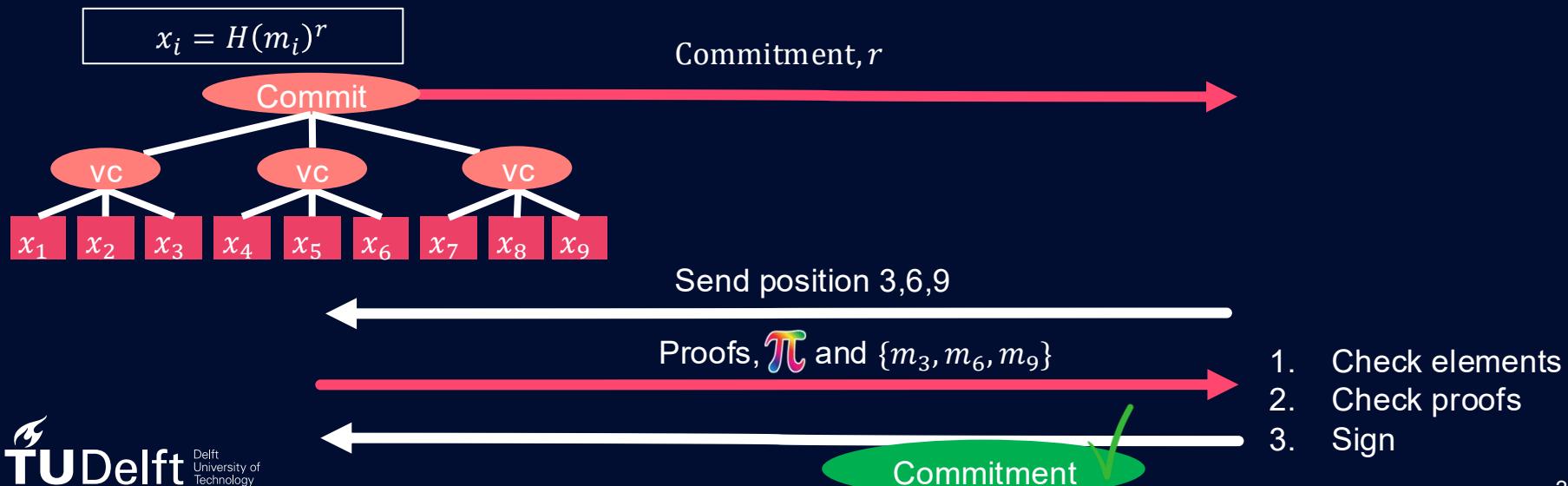
$x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \quad x_7 \quad x_8 \quad x_9$

Make Verkle tree

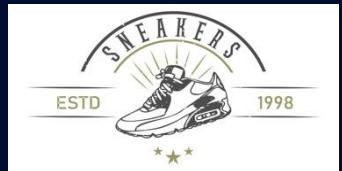
# Authorization – math version



$\{m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, m_9\}$



# Intersection – math version



$\{m_1, m_2, m_3, m_4,$   
 $m_5, m_6, m_7, m_8, m_9\}$

$x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \quad x_7 \quad x_8 \quad x_9$



Commitment



1. Build tree
2. Check commitment

$\{\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_8, \mu_7\}$

$$x_i = H(m_i)^r$$

$$y_i^r = H(\mu_i)^{sr}$$

Intersect

$$\hat{x}_i = H(m_i)^{rs}$$

$\{y_i\}$

$$y_i = H(\mu_i)^s$$

$\{\hat{x}_i\}$

$$\hat{x}_i = x_i^s$$

# Secure

- Judge cannot learn more than the revealed elements
  - Store cannot learn more than the intersection
  - Google cannot learn any element
- 
- Store: no universe attack
  - Store: only intersection with authorized elements



00

00

00

Commitment

$$y_i = H(m_i)^s$$

$$x_i = H(m_i)^r$$

$m_i$  for several  $i$

$x_i = H(m_i)^{rs}$

$r$

# Bandwidth efficient

## Our protocol

### Authorization

- Commitment
- Indices
- $\pi$
- Commitment ✓

### Intersection

- $x_i = H(m_i)^r$
- $y_i = H(m_i)^s$
- $x_i = H(m_i)^{rs}$

## Falzon & Markatou

### Authorization

- $x_1$   $x_2$   $x_3$   $x_4$   $x_5$   $x_6$   $x_7$   $x_8$   $x_9$
- Indices
- Proofs
- $x_1$  ✓  $x_2$  ✓  $x_3$  ✓  $x_4$  ✓  $x_5$  ✓  $x_6$  ✓  $x_7$  ✓  $x_8$  ✓  $x_9$  ✓

### Intersection

- $y_i$
- $y_i$
- $y_i$

# It works!

- Correct
- Secure
- Bandwidth efficient



# Last question

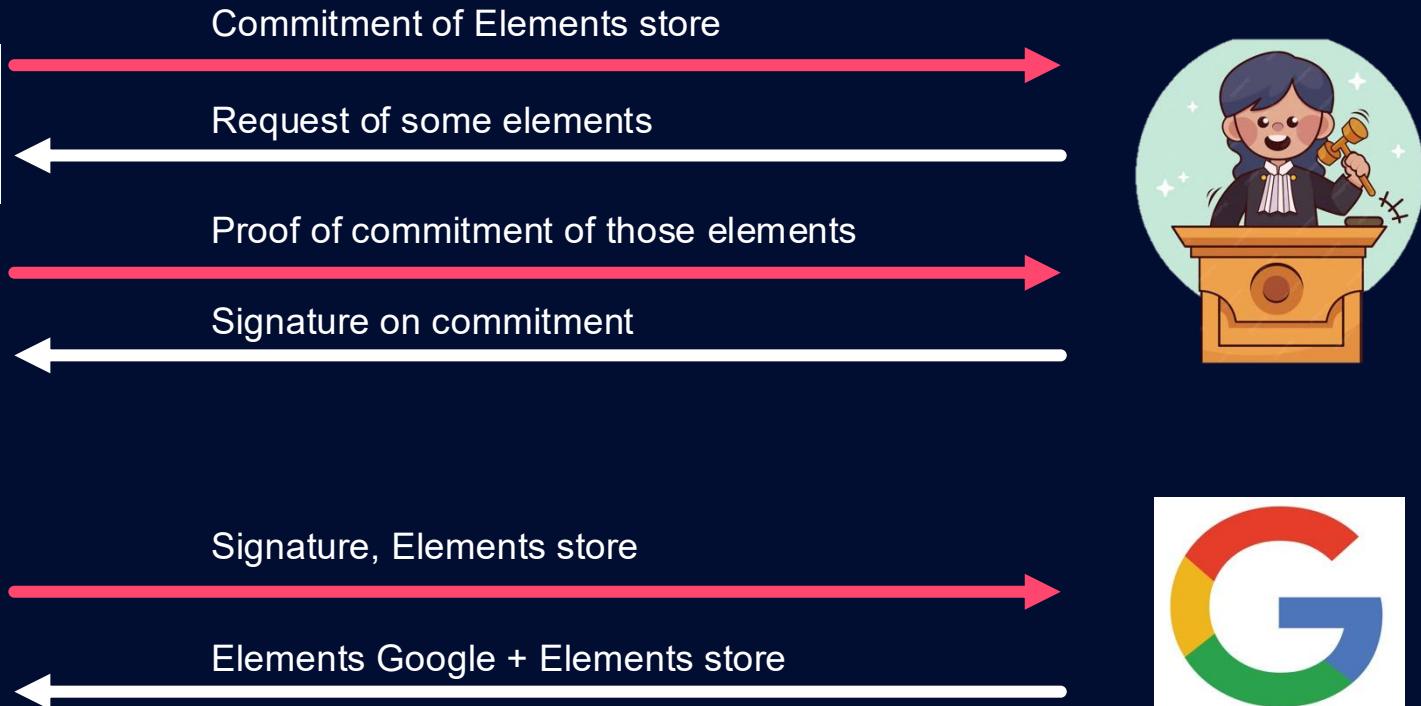
10%

How many elements should  
the judge check?

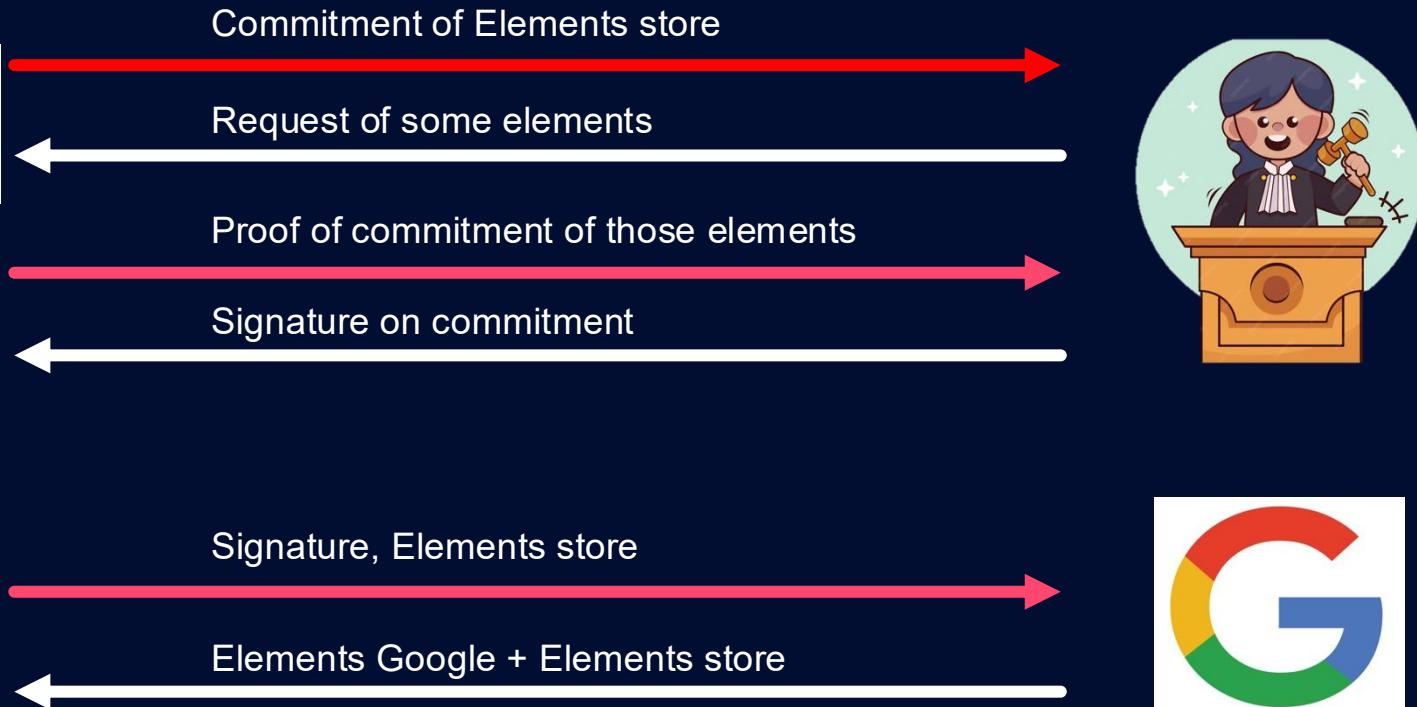
???

30%

# Where can we cheat? And not get caught...



# Where can we cheat? And not get caught...



# Game theoretic analysis; where can we cheat?

- Input  $\{m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, \textcolor{red}{m_{666}}, m_n\}$
- Judge samples  $p$  percentage of elements
- Let  $\mu$  be percentage malicious elements
- Probability Judge sees  $m_{666}$ ?  
$$\binom{n-n\mu}{np}$$
- 
- Here, for  $np = 1$ , chance of seeing  $\textcolor{red}{m_{666}} = 10\%$

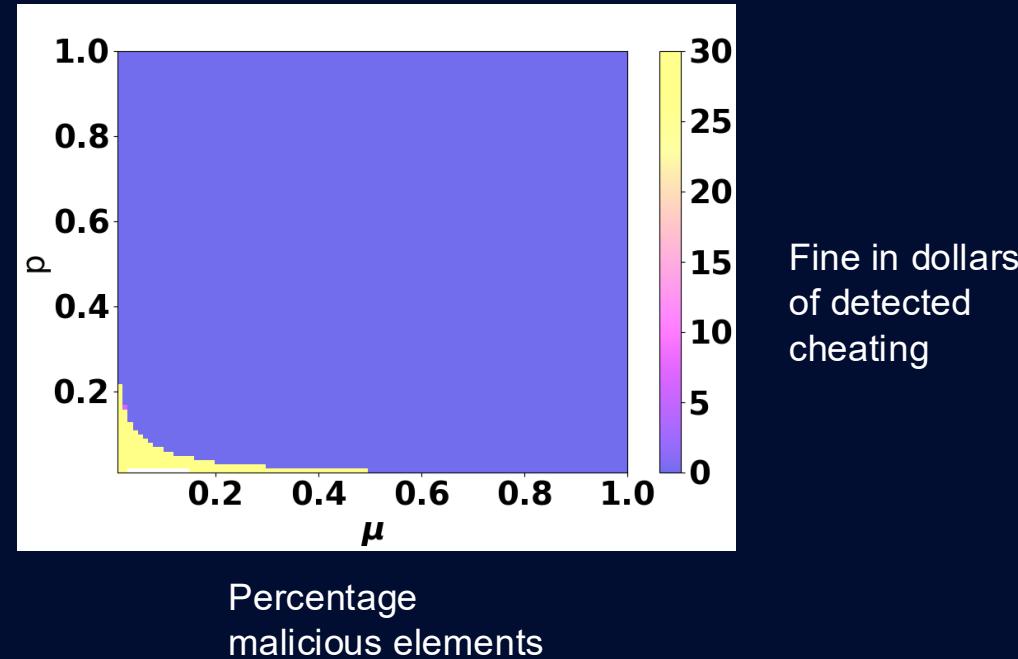
# Is it worth cheating?

- Loss of revealing elements to judge
- Gain of computing intersection
- Gain of finding malicious elements
- Loss of getting detected
- $\$20 p \cdot n$
- \$353
- $\$20 \mu \cdot n$
- ????? fine



# Cost of fine to ensure Store behaves well

Percentage  
the judge  
checks



# Runtime protocol – green is where we do better

Set sizes	Scheme	Communication (KB)		Runtime (ms)			Total Runtime			
		Auth	Inter	Judge	Store	Google	LAN	1Gbps	200Mbps	50Mbps
Google	Store									
$2^{10}$	FM	147	202473	1	750	8332	9083	9643	9643	9643
	Verkle	9	4203	21	58	124	202	203	203	203
$2^{20}$	$2^{16}$	FM	9520	208666	78	1301	8361	9741	10301	10301
	$2^{20}$	Verkle	557	4719	37	239	162	438	438	439
$2^{20}$	$2^{20}$	FM	152256	303038	1196	10558	8390	20145	20705	20705
	$2^{20}$	Verkle	8907	12583	168	2263	979	3410	3411	3412
$2^{24}$	FM	2436770	1812987	19454	171482	8357	199295	199855	199855	199855
	Verkle	142499	138412	1382	35356	11076	47814	47817	47826	47859
$2^{26}$	FM	9738788	6644826	75754	727536	8399	811690	812250	812250	812252
	Verkle	569989	541065	4832	166082	40176	211090	211099	211135	211268

# What did we do today?

- Are we ready for the PQ-era? - No
- How to prevent malicious input in PSI?
  - Solution Falzon Markatou – Partial Authorized PSI
  - Our solution – faster
  - Determine percentage  $p$  – game theory
- Future research
  - Post-Quantum secure
  - Eliminate the judge



Receive results PQC readiness



Check out our eprint

Tjitske Koster  
[t.o.koster@tudelft.nl](mailto:t.o.koster@tudelft.nl)

Contact me